Nutzen und Kosten von serviceorientierten Architekturen

Inauguraldissertation
zur Erlangung des Doktorgrades
der Wirtschafts- und Sozialwissenschaftlichen Fakultät
der Universität zu Köln

2006

vorgelegt
von Diplom-Wirtschaftsinformatiker
Kai Jan Oey
aus Tübingen

Nutzen und Kosten von serviceorientierten Architekturen

Inauguraldissertation
zur Erlangung des Doktorgrades
der Wirtschafts- und Sozialwissenschaftlichen Fakultät
der Universität zu Köln

2006

Referent:
Professor Dr. Werner Mellis

Korreferent:
Professor Dr. Dietrich Seibt

Tag der Promotion:
21. Dezember 2006

vorgelegt
von Diplom-Wirtschaftsinformatiker
Kai Jan Oey
aus Tübingen

Für Anke

und

Brigitte und Jan, meine Eltern.

Aus rechtlichen Gründen ist darauf hingewiesen, dass die in dieser Arbeit verwendeten Namen von Firmen, Organisationen und Produkten möglicherweise die Warenzeichen der jeweiligen Besitzer sind.

Inhaltsübersicht

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

BPM.....Business Process Management

CSV .....Comma Separated Values

EAI.....Enterprise Application Integration

EDA .....Event-driven Architecture

ERM.....Entity-Relationship-Model

HTTP .....Hypertext Transport Protocol

IBM.....International Business Machines

IEEE.....Institute of Electrical and Electronics Engineers

IS.....Informationssystem

ISO.....International Organization for Standardization

IT .....Informationstechnologie (Information Technology)

KPI.....Key Process Indicators

MDA.....Model-driven Architecture

QFD .....Quality Function Deployment

SAP.....Systeme, Anwendungen, Produkte in der Datenverarbeitung

SOA .....Serviceorientierte Architektur (Service-oriented Architecture)

UML .....Unified Modeling Language

XML .....Extensible Markup Language

1. Einleitung

Die Diskussion um die Gestaltung der betrieblichen Datenverarbeitung wird gegenwärtig durch ein Schlagwort geprägt: serviceorientierte Architekturen, kurz SOA. Marktstudien prognostizieren für die nächsten Jahre einen rasanten, branchenübergreifenden Zuwachs der Unternehmen, die eine SOA einsetzen. Die wachsende Bedeutung dieses Architekturkonzepts wird von einigen Marktforschungsinstituten einem Paradigmenwechsel gleichgesetzt – vergleichbar mit objektorientierten Ansätzen, die bereits in den sechziger Jahren entwickelt wurden.[1]

Die traditionelle Sichtweise auf eine Unternehmens-IT ist, dass für spezialisierte Aufgaben innerhalb eines Unternehmens spezialisierte IT-Systeme genutzt werden.[2] Es wird die Meinung vertreten, dass IT-Systeme für verschiedene Hierarchieebenen und für verschiedene funktionale Bereiche eines Unternehmens eingesetzt werden sollen. Aus solchen Vorgehensweisen zur Gestaltung einer Unternehmens-IT entstanden die sog. Legacy[3]- oder auch Stovepipe[4]-Systeme. Die Folge ist, dass gegenwärtig mehrere eigenständige IT-Systeme innerhalb eines Unternehmens eingesetzt werden. Jedes dieser IT-Systeme unterstützt einen abgegrenzten Teil eines Unternehmens – unterschieden nach strukturellen oder funktionalen Kriterien.[5]

Die gegenwärtige Sichtweise auf eine Unternehmens-IT ist weiterhin, dass eine Unternehmens-IT als ein umfassendes IT-System angesehen werden muss, das in der Lage ist, alle Prozesse und Daten des Unternehmens zu unterstützen und zu verwalten.[6] Dieser Aspekt ist geprägt von dem Ausdruck der Enterprise Application Integration, kurz EAI.[7] EAI steht für die IT-Strategie, bei der alle Softwaresysteme eines Unternehmens einem einheitlichen Aufbau folgen. Ein Grund für diese Entwicklung ist, dass Unternehmen unter hohem wirtschaftlichen Druck stehen und in ihrer Unternehmens-IT Po-

---

[1] Vgl. CapGemini /IT-Trends 2005/ S. 4-6; Gruhn, Thiel /Komponentenmodelle/ S. IX.

[2] Vgl. zum folgenden Absatz Laudon, Laudon /Business/ S. 40-44; Ferstl, Sinz /Grundlagen/ S. 196.

[3] Vgl. Sommerville /Software Engineering/ S. 589-606; Inmon, Zachman, Geiger /Data/ S. 33.

[4] Linthicum /Application Integration/ S. 11-14.

[5] Vgl. beispielsweise Moormann für die Situation im Finanzsektor. Vgl. Moormann /Umbruch/ S. 11-12.

[6] Vgl. Ferstl, Sinz /Wirtschaftsinformatik/ S. 196-200.

[7] Vgl. weiterführend zur EAI Linthicum /Next Generation/; Keller /EAI/; Aier, Schönherr /Flexibilisierung/; Kaib /EAI/.

tential zur Effizienzsteigerung durch EAI sehen.[8] Eine Unternehmens-IT kann mit Hilfe verschiedener Methoden integriert werden.[9] Geeignet erscheint eine Integration auf Softwarearchitekturebene, weil diese den Gesamtaufbau einer Unternehmens-IT definiert und die Grundlage der Unternehmens-IT darstellt.

Die Unternehmens-IT ist auch extrem wichtig geworden, weil sie Produktion (z. B. durch Automatisierung) und Produkte (z. B. durch Komplementärleistungen) des Unternehmens unterstützt. Teilweise ist die Unternehmens-IT sogar der wichtigste Produktionsfaktor eines Produktes – z. B. im Finanzsektor. Deshalb steht die Entwicklung einer Unternehmens-IT unter hohem wirtschaftlichem Druck.[10] Die Realisierung von wirtschaftlich relevanten Potentialen, z. B. Wettbewerbsfähigkeit, Umsatz und Margen, ist nicht nur durch den Einsatz einer Unternehmens-IT möglich, sondern auch durch Verbesserungen der Softwaretechnik:[11] Dazu zählen Programmiersprachen, Softwareentwicklungsvorgehen und Softwarearchitekturen.

Der Fokus dieser Arbeit liegt auf den Softwarearchitekturen. Diese geben einheitliche Vorgehensweisen und Strukturen für eine Unternehmens-IT vor. Durch die verschiedene Ausgestaltung der durch Softwarearchitekturen determinierbaren Charakteristika sollen bestimmte Wirkungen auf die gesamte Unternehmens-IT und deren Eigenschaften ausgeübt werden – Wirkungen, die wirtschaftlich relevante Potentiale des Unternehmens wesentlich verbessern sollen.

Diese Arbeit beschäftigt sich mit dem Wirtschaftlichkeitspotential von Softwarearchitekturen anhand des Konzeptes SOA. Für SOA wird eine Wirtschaftlichkeitsanalyse durchgeführt, die die oben angesprochenen Wirkungen untersucht. Die ‚Idee SOA' geht bis in das Jahr 1993 zurück, in dem z. B. Hammer und Champy[12] schon den Ansatz der Dezentralisierung und Vernetzung betrieblicher Informationssysteme vorschlagen. Der

---

[8] Z. B. wird von dem Potential zur Komplexitätsreduktion, zur Schaffung eines optimalen Kopplungsgrads oder von unternehmensweiter Wiederverwendung gesprochen. Vgl. Linthicum /Application Integration/ S. 9; Kaib /EAI/ S. 22-34; Cummins /Enterprise Integration/ S. 23-44; Themistocleous, Irani /Benchmarking/ S. 325-328; Raasch /Systementwicklung/ S. 43-44; Ruh, Maginnis, Brown /EAI/ S. 3-7, S. 12, S. 20 & S. 156.

[9] Vgl. Aier, Schönherr /Flexibilisierung/ S. 16.

[10] Vgl. Schreyögg /Organisation/ S. 305-310.

[11] Z. B. werden sehr kurze Entwicklungszeiten angestrebt oder vorgegeben, um wettbewerbsfähig bleiben zu können. Vgl. Brandt-Pook u. a. /Anwendungsentwicklung/ S. 249; Mellis /Process/ S. 2. Zu Vorgehensmodellen vgl. beispielsweise Boehm /Spiral Model/, Seibt /Vorgehensmodell/ und für eine Übersicht Mellis /Softwaremanagement/ S. 79-110.

Begriff SOA wurde erstmals 1996 von Gartner verwendet.[13] Populär wurde dieses Konzept jedoch erst ca. fünf Jahre später mit der Einführung von Technologien, die es ermöglichten, das Konzept umzusetzen: Web Services und moderne Middleware.[14] Belegt wird dies durch die Bekanntgaben einiger renommierter Unternehmen, wie z. B. IBM und SAP, zukünftig auf SOA bei der Entwicklung von IT-Systemen zu setzen.[15] Der Trend lässt sich auch anhand der zunehmenden Anzahl von Veröffentlichungen über SOA, wie z. B. im Cutter IT Journal, belegen.[16] Weil SOA in Grundzügen auf bekannten Konzepten aufbaut und wirtschaftliche Ziele angestrebt werden, die in der ‚IT-losgelösten Welt' schon lange relevant sind, wird SOA oft mit der Kritik konfrontiert, nicht mehr als ‚alter Wein in neuen Schläuchen' zu sein.[17] In dieser Arbeit wird untersucht, inwieweit SOA den heute gestellten Anforderungen und wirtschaftlichen Zielen gerecht wird.

Im Folgenden wird die Problemstellung der Arbeit dargestellt (Kapitel 1.1). Anschließend werden die Zielsetzung (Kapitel 1.2) und das Vorgehen der Arbeit (Kapitel 1.3) vorgestellt. Abschließend werden Begriffe geklärt (Kapitel 1.4).

1.1 Problemstellung

Ziel der Forschungsarbeit ist es, der Praxis wertvolle Hinweise geben zu können, welches Wirtschaftlichkeitspotential in einer SOA steckt. Grundlage der Arbeit sind Fallstudien aktueller Projekte, in denen SOA umgesetzt wird. Des Weiteren werden veröffentlichte empirische und konzeptionelle Beiträge herangezogen. Eine qualitative empirische Erhebung sichert den Erkenntnisgewinn der Arbeit ab.

Um Bewertungen von Softwarearchitekturen vornehmen zu können, müssen Analysen zur Softwarearchitekturbewertung laufend erweitert und angepasst werden.[18] Dies liegt darin begründet, dass der Gegenstand der Bewertungsanalyse, die Softwarearchitektu-

---

[12] vgl. Hammer, Champy /Reengineering/ S. 92-101.

[13] Vgl. Natis /SOA Scenario/ S. 1; Natis /SOA/ S. 1.

[14] Vgl. Dostal, Jeckle /Service-orientierte Architektur/ S. 53; Carlson, Tyomkin /Good Design/ S. 14.

[15] Vgl. beispielsweise Mitteilungen der Computer Zeitung und Computer Woche aus den Jahren 2003-2006.

[16] Vgl. Cutter IT Journal, Mai 2004 (Vol. 17, Nr. 5) mit dem Schwerpunkt „Service Orientation".

[17] Vgl. Allen /Statement/ S. 2-4.

[18] Vgl. zum folgenden Absatz Carrière, Kazman, Woods /Architectural Quality/ S. 1.

ren, vor allem Systeme betrifft, die eine lange Lebensdauer haben, damit ständig wachsen und einem Wandel unterworfen sind.

Diese Arbeit stellt eine Wirtschaftlichkeitsanalyse für die spezielle Softwarearchitektur SOA vor. Bei der Erstellung von Software kann nicht eine bestimmte Regel aufgestellt werden, die alle Ziele erfüllt, die bei der Entwicklung von Software verfolgt werden können,[19] vielmehr existieren verschiedene Vorgehensweisen z. B. bei der Entwicklung und Strukturierung von Software, die bestimmten Zielen mehr oder weniger gut gerecht werden. Ein Bereich, in dem alternative Gestaltungen einer Software vorgenommen werden können, ist die Softwarearchitektur.

Das hier gezogene Ergebnis ist unter Beachtung allgemeiner Vorgehensweisen richtig. Für konkrete Ausprägungen einer SOA – also für Projekte in der Praxis – kann nicht allgemeingültig argumentiert werden. Konkrete Anwendungen einer SOA müssen eine Bewertung anhand des vorgestellten Faktorenmodells durchführen, um zu einem individuell gültigen Ergebnis zu gelangen.

Die Problemstellung der Arbeit wird dargestellt anhand des untersuchten Forschungsproblems (Kapitel 1.1.1) und dessen Signifikanz (Kapitel 1.1.2).

1.1.1 Praxis- und Forschungsproblem

Ein Forschungsproblem steht in Beziehung zu einem Praxisproblem.[20] Ein *Praxisproblem* wird durch einen Zustand in der Welt verursacht, der nachteilig für den Betrachter ist. Ein Praxisproblem wird gelöst, indem der Betrachter etwas unternimmt, um den Zustand zu ändern. Beim Durchlaufen dieses Prozesses wird der Betrachter motiviert Fragen zu stellen, wie der Zustand geändert werden kann: sog. Forschungsfragen. Diese Fragen definieren das *Forschungsproblem*, welches gelöst werden muss, um das Praxisproblem (möglichst positiv) zu beeinflussen. Ein Forschungsproblem wird motiviert, weil ein Zustand der Welt falsch oder nicht vollständig verstanden ist. Es ändert die Welt nicht, sondern erhöht beziehungsweise verbessert das Verständnis des Zustands. Die Lösung des Forschungsproblems erzeugt *Forschungsantworten*. Diese liefern dem

---

[19] Vgl. Boehm /Economics/ S. 21-23.

[20] Vgl. zum folgenden Absatz Booth, Colomb, Williams /Craft/ S. 57-60.

Betrachter die Informationen, damit er die Aktivitäten durchführen kann, die helfen, das Praxisproblem zu lösen.

Im Folgenden wird dargestellt, wie das Forschungsproblem dieser Arbeit abgeleitet wird.

Praxisproblem

In Unternehmen liegen typischerweise heterogene Systemlandschaften vor.[21] In den 1990er Jahren beginnend fand die Integration immer komplexer werdender Unternehmens-IT-Systeme auf Basis von individualentwickelten Punkt-zu-Punkt Integrationen statt.[22] Die Komplexität der Unternehmens-IT erhöhte sich daraufhin um ein Vielfaches, weil eine integrierte Behandlung von Anforderungen, IT-Systemen und Kommunikationstechniken erforderlich ist.[23] Änderungen eines Systems konnten nicht einfach vorgenommen werden, ohne dass andere Systeme direkt betroffen waren und ebenfalls geändert werden mussten. Die Probleme der Softwareentwicklung – Zeitdruck in der Entwicklung, Qualitätsprobleme, steigende Kosten, verringerte Wirtschaftlichkeit usw. – verstärkten sich.

Gleichzeitig wird festgestellt, dass sich die unternehmensweite Integration der Softwaresysteme nicht auf die Betrachtung der technischen Komponenten beschränken lässt.[24] Vielmehr ist es ebenso notwendig, die Interdependenzen zwischen organisatorischen Gestaltungsaspekten und der Unternehmens-IT zu berücksichtigen.[25] Grundlegend für den wirtschaftlichen Erfolg des Unternehmens ist die Fähigkeit, flexibel auf Trends und sich verändernde Rahmenbedingungen reagieren zu können.[26] Deswegen muss eine moderne Unternehmens-IT auf den ständigen Wandel des Unternehmens und

---

[21] Vgl. Kapitel 1.

[22] Vgl. Aier, Schönherr /Flexibilisierung/ S. 12-13.

[23] Vgl. Willgosch /Informationslieferanten/ S. 310.

[24] Vgl. Aier, Schönherr /Flexibilisierung/ S. 5; Schon 1979 sprach Szyperski von einem siamesischen Zwillingscharakter: „Die Interdependenzen werden deutlich durch die Tatsache, dass ohne Kommunikation keine Information möglich ist und umgekehrt. Es ist ratsam, sie ihrem siamesischem Zwillingscharakter entsprechend miteinander verbunden zu begreifen.“ Vgl. Szyperski /Strategisches Informationsmanagement/ S. 3.

[25] Vgl. Aier, Schönherr /Flexibilisierung/ S. 10-11.

[26] Vgl. Baumöl, Winter /Qualifikation/ S. 46-47; Hahn, Hungenberg /Controllingkonzepte/ S. 109; Horvath /Controlling/ S. 3-6; Krystek /Vertrauen/ S. 276-278; Karch u. a. /SAP/ S. 19.

seines Umfeldes vorbereitet sein.[27] Z. B. wird es für eine Unternehmens-IT immer wichtiger, über Unternehmensgrenzen hinweg andere Systeme anzubinden und zu integrieren[28]. Auch muss eine Unternehmens-IT heute öfters in der Lage sein, ein Unternehmen global zu unterstützten.[29]

Unternehmen haben realisiert, dass ein ständiger Wandel ihrer Unternehmens-IT notwendig ist, um am Markt bestehen zu können. Die (Weiter-)Entwicklung der Unternehmens-IT-Systeme, um von Zeitersparnisse, Produktivitäts- oder Qualitätssteigerungen durch die IT zu erzielen, ist jedoch teuer und zeitaufwändig. Dadurch wird zum einen der operative Nutzen der Unternehmens-IT geschmälert. Zum anderen wird der unternehmerische Nutzen der Unternehmens-IT geschmälert. Mögliche Umsatzsteigerungen durch eine verbesserte Unterstützung oder eine erhöhte Kundenzufriedenheit durch günstigere Preise können nicht erreicht werden.

Forschungsfrage

Diese Probleme motivieren die Forschungsfrage, wie eine Unternehmens-IT implementiert werden sollte, damit sich maximaler unternehmerischer Nutzen ergibt – sowohl in der Entwicklung und Wartung der Unternehmens-IT als auch in der Unterstützung des Unternehmens.[30]

Diese Frage nach wirtschaftlichem Nutzen durch IT allgemeingültig zu beantworten wird nicht möglich sein. Zum einen, weil sich Unternehmen unterscheiden, in unterschiedlichen Märkten mit unterschiedlichen Kostenstrukturen agieren und dadurch verschiedene Anforderungen an ihre Unternehmens-IT haben; zum anderen, weil sich eine hohe Vielfalt an Alternativen bezüglich der verwendbaren Technologien und Vorgehensweisen ergibt.

---

[27] Vgl. Sommerville /Software Engineering/ S. 29; Karch u. a. /SAP/ S. 11; Raasch /Systementwicklung/ S. 43-44.

[28] Karch u. a. /SAP/ S. 20.

[29] Vgl. Frese /Organisation/ S. 128-129; Moormann /Umbruch/ S. 5-6, Hahn /Strategische Führung/ S. 1038; Seibt /Begriffe und Aufgaben/ S. 25.

[30] An dieser Stelle sei darauf hingewiesen, dass die ‘Forschungsfrage’ nicht die Frage darstellt, die in dieser Arbeit beantwortet wird. Diese Frage führt vielmehr zu dem Forschungsproblem, das anhand von konkreten Fragenstellungen beantwortet wird.

Forschungsproblem

Wie oben aufgezeigt, lässt sich die Forschungsfrage auf die Softwarearchitektur eingrenzen. Eine solche Eingrenzung erhöht und verbessert das Verständnis der aufgeworfenen Forschungsfrage.

Die Evaluation von Kosten und Nutzen von Informationssystemen ist kein neues Problem.[31] Dies gilt speziell für die Bewertung der Kosten und des Nutzens von Softwarearchitekturen. Zachman stellt fest, dass es nicht möglich ist, Softwarearchitekturen kostenzentriert zu bewerten, weil sie keine expliziten Kosten darstellen.[32] Doch werden Kosten und Nutzen technischer Aspekte als Argumente bevorzugt, wenn architektonische Verbesserungen diskutiert werden.[33] Eine technische Sicht auf Softwarearchitekturen ist jedoch nur eine Sicht – eine weitere ist die wirtschaftliche.[34] Eine Wirtschaftlichkeitsanalyse einer Unternehmens-IT lässt sich nicht losgelöst von Fragen der Unternehmensorganisation betrachten, weil zwischen ihnen komplexe Wechselwirkungen und Abhängigkeiten bestehen.[35] Ferner wirkt sich eine Softwarearchitektur langfristig auf den Erfolg eines Informationssystems aus.[36] Dies sind Probleme, die durch die Beantwortung der Frage *‚Verbessert SOA die Wirtschaftlichkeit einer Unternehmens-IT?*' bewältigt werden sollen.

Die Zielsetzung (Kapitel 1.2) unterteilt diese Frage in die konkreten Fragestellungen, die in dieser Arbeit beantwortet werden.

Forschungsantwort und ‚Lösung' des Praxisproblems

Die Forschungsantwort und die Lösung des Praxisproblems werden – soweit eine Beantwortung des Forschungsproblems einwandfrei möglich ist – im Resümee und Ausblick gegeben.

---

[31] Vgl. Smithson, Hirschheim /Old Problem/ S. 160-165; Shin /Strategic Choice/ S. 227-229; Irani, Love /Frame of Reference/ S. 74-75; Potthof /Empirische Studien/ S. 58-63.

[32] Vgl. Zachman /Cost-Justify Architecture/ S. 12.

[33] Vgl. Rothman /Architectual Infrastructur/ S. 15-19; Hohmann /Beyond/ S. 52-52.

[34] Vgl. Rothman /Architectual Infrastructur/ S. 15-19; Hohmann /Beyond/ S. 52-52.

[35] Vgl. Frese /Kommunikationseffekte/ S. 200-203.

[36] Vgl. Hohmann /Beyond/ S. 4-7; Reinertsen /System Architecture/ S. 20-25.

1.1.2 Signifikanz der Problemstellung

Die Signifikanz der Problemstellung kann auf zwei Ebenen verdeutlicht werden: Die Signifikanz des Praxisproblems und die des Forschungsproblems.

Signifikanz des Praxisproblems:

Unternehmen können heute schwerlich auf eine Unternehmens-IT verzichten.[37] Eine Unternehmens-IT wird unternehmensweit eingesetzt.[38] Von ihr sind Unternehmen sogar abhängig.[39] Die Nutzung von Software alleine ist gegenwärtig zu einem Allgemeingut geworden[40] und eine Nicht-Nutzung kann sich negativ auf die Wettbewerbsfähigkeit von Unternehmen auswirken.[41] Die Nutzung von Informations- und Kommunikationstechnologien und folglich auch die Nutzung einer Unternehmens-IT, sind ein überlebensnotwendiger Wettbewerbsfaktor geworden.[42] Z. B. kann hier die Luftfahrtbranche angeführt werden: Der Trend bezüglich IT-Systeme ist, dass Luftfahrt-Allianzen eine einheitliche unternehmensübergreifende IT-Infrastruktur anstreben.[43] Aktuell baut z. B. die Star Alliance eine gemeinsame IT-Plattform (die sog. ‚Common IT-Platform').[44] Zudem fanden Firmenübernahmen in der Luftfahrtindustrie, die Auswirkungen auf die entsprechenden Unternehmens-IT-Systeme haben, vermehrt statt: 17 Zusammenschlüsse bzw. Übernahmen innerhalb von vier Jahren.[45]

Eine Unternehmens-IT baut auf einer Softwarearchitektur auf. Die Softwarearchitektur stellt Gestaltungsrichtlinien für die Strukturierung einer Unternehmens-IT auf und be-

---

[37] Vgl. Mellis /Projektmanagement/ S. 2; Lucas /Information Technology/ S. 13; Horváth /Controling/ S. 675.

[38] Vgl. Lewin, Hunter /Information Technology/ S. 256.

[39] Vgl. Sommerville /Sofware Engineering/ S. 609; Gronau /Wandlungsfähige Informationssystemarchitekturen/ S. 216; für ein konkretes Beispiel einer Branche vgl. Buhalis /eAirlines/ S. 811-814.

[40] Vgl. Carr /Software/ S. 271-273; Kieser, Walgenbach /Organisation/ S. 400-405.

[41] Vgl. Berensmann /IT matters/ S. 275-276.

[42] Vgl. Szyperski /Strategisches Informationsmanagement/ S. 7-8. Andere Autoren erweitern diesen Standpunkt dadurch, dass sich langfristig eine Gleichbedeutung von IT und Unternehmen ergeben wird. Vgl. Kieser, Walgenbach /Organisation/ S. 384; Morgan /Bilder/ S. 117 nach Aier, Schönherr /Flexibilisierung/ S. 5; Festgestellt wurde, dass die Unternehmens-IT in vier Wirkungsrichtungen unterschieden werden kann: als Fabrik, als Waffe, zur Unterstützung oder zum Durchbruch. Vgl. Macharzina /Unternehmensführung/ S. 650-652.

[43] Vgl. Baker /Common Cause/ S. 46-47.

[44] Vgl. Baker /Common Cause/ S. 46-47.

[45] Vgl. Field /Pain relief/ S. 27.

einflusst die Wirtschaftlichkeit einer Unternehmens-IT.[46] Eine Softwarearchitektur hat Einfluss auf die Fähigkeiten einer Unternehmens-IT und insofern auf die Fähigkeiten eines Unternehmens: Z. B. empfiehlt sich heute die simultane Entwicklung von Prozessorganisation und Unternehmens-IT, um die Potentiale der Unternehmens-IT bestmöglich nutzen zu können.[47] Gleichzeitig sind Unternehmen in sich wandelnden Unternehmensumfeldern eingebettet[48] und müssen aktiv handeln, um am Markt bestehen zu können.[49]

Nach einer weltweiten Studie der Gartner Group aus dem Jahr 2003 betragen die Fehlinvestitionen im IT-Bereich etwa 20 % der gesamten IT-Ausgaben.[50] Zu den Ursachen gehört die Unfähigkeit der Verantwortlichen, diejenigen Projekte auszuwählen, die für das investierende Unternehmen Nutzen generieren. Meistens werden bei der Evaluierung von Informationssystemen Methoden der klassischen Investitionsrechnung eingesetzt.[51] Dabei ergeben sich Probleme, weil eine Bewertung oft nicht vollständig monetär durchführbar ist. Die Folge ist, dass bei der Evaluierung ein Großteil des realisierbaren Nutzens ignoriert wird oder die systematische Erfassung dieses Nutzens durch die subjektive Einschätzung des Managements ersetzt wird.[52] In letzterem Fall wird von „acts of faith“[53] gesprochen, d. h., ein Entscheider ist der Meinung, dass ein bestimmtes Informationssystem von hoher strategischer Bedeutung ist und entsprechenden Nutzen liefern wird. Für Unternehmen ist die Lösung des Praxisproblems – auch die Steigerung des Wissens über das Praxisproblem – zur Unterstützung der Bewertung von Unternehmens-IT-Systemen von großer Relevanz.

---

[46] Vgl. Kapitel 2.1.

[47] Vgl. Macharzina /Unternehmensführung/ S. 669-670, 736.

[48] Vgl. Krüger /Management/ S. 228-236.

[49] Vgl. Macharzina /Unternehmensführung/ S. 199-200.

[50] Vgl. zu diesem Absatz Schwab /IT-Strategie/.

[51] Vgl. Fitzgerald /Evaluating/ 17; Dobschütz /Wirtschaftlichkeitsanalyse/ 42; Irani und Love stellen zusätzlich auch fest, dass hier häufig von Standardvorgehen abgewichen wird, da die Investition in IT/IS als Konsumierung und nicht als Investition angesehen wird (vgl. Irani, Love /Frame of Referenze/ S. 74); zudem dauere eine Bewertung zu lange und es müssten zu viele Beteiligte mit ihren jeweiligen Interessen beachten werden. Dies wird als zu teuer angesehen. Vgl. Irani, Love /Systems Evaluation/ S. 183.

[52] Fitzgerald /Evaluating/ S. 15-16; Pietsch /Bewertung/ S. 12.

[53] Vgl. Walter, Spitta /Evaluation/ S. 171; Coleman /Value/ S. 59; Farbey, Targett, Land /Assess/ S. 57.

Signifikanz des Forschungsproblems:

Gestaltungsalternativen in der Wirtschaftsinformatik müssen bezüglich der Wirtschaftlichkeit des jeweiligen Einsatzes untersucht werden. Sie stellt das Grundprinzip ökonomischen Handelns dar.[54] Wirtschaftlichkeit wird je nach Beurteilungsgegenstand definiert, stellt monetär betrachtet jedoch grundlegend ein Einnahmen-Ausgaben-Verhältnis (weniger stark monetär betrachtet ein Leistungs-Mitteleinsatz-Verhältnis) dar. Bezüglich der Entwicklung von Software ist das Ziel vor allem die Steigerung des Nutzens durch den Einsatz von IT-Systemen.[55]

Die Entwicklung von Softwarearchitekturen gilt als eine der Hauptaufgaben der Wirtschaftsinformatik.[56] Die Untersuchung der Wirtschaftlichkeit von Softwarearchitekturen ist freilich sowohl für die Wissenschaft als auch für die Praxis von hoher Relevanz. Die Untersuchung der Wirtschaftlichkeit einer SOA ist eine spezielle Untersuchung der Wirtschaftlichkeit einer Softwarearchitektur. In der Praxis zeigt sich, dass das Forschungsproblem deswegen relevant ist, weil z. B. 2005 bereits 7 % der Unternehmen in der Luftfahrtbranche SOA einsetzten.[57] Weitere 29 % planen, diese innerhalb der kommenden zwei Jahre einzuführen.

In dieser Arbeit wird eine ex-ante Analyse der Wirtschaftlichkeit von SOA vorgenommen. Ex-ante Analysen sind in der Praxis von Interesse, weil sie für Investitionsentscheidungen notwendig sind.[58] Die ex-ante Analysen der Wirtschaftlichkeit, die durch den Einsatz von Informationssystemen entsteht, gestaltet sich im Vergleich zur reinen Kostenbewertung jedoch besonders schwierig.[59] Auf Basis der Wirtschaftlichkeitsbetrachtung kann die Praxis, z. B. Beratungsunternehmen und große Konzerne mit eigener IT-Abteilung, bei der Bearbeitung dieser Themen auf Erkenntnisse der Wissenschaft zurückgreifen.

---

[54] Einen Überblick über verschiedene Auffassungen von Wirtschaftlichkeit gibt Antweiler /Wirtschaftlichkeitsanalyse/ S. 56-62.

[55] Vgl. Aier, Schönherr /Flexibilisierung/ S. 14.

[56] Vgl. Seibt /Probleme und Aufgaben/ S. 14.

[57] Vgl. zum folgenden Absatz Baker /IT Trends/ S. 40. An der Studie haben ca. 50 % der Top 200 Lufttransportunternehmen teilgenommen.

[58] Vgl. Irani, Love /Frame of Reference/ S. 74-76.

[59] Vgl. Schumann /Wirtschaftlichkeitsbeurteilung/ S. 168; Nagel /Nutzen/ S. 29; Renkema, Berghout /Methodologies/ S. 1. Vgl. auch Kapitel 3.2.1.

Umfassende Untersuchungen, über die Wirtschaftlichkeit einer SOA, existieren nicht.[60] Diese Arbeit nimmt diesen Mangel zum Anlass, die Wirtschaftlichkeit zu untersuchen, die durch den Einsatz von SOA entstehen kann. Der Untersuchungsgegenstand ist somit die Wirtschaftlichkeitsanalyse der Softwarearchitektur SOA. Die Untersuchung, welchen Einfluss eine SOA auf die Wirtschaftlichkeit eines Unternehmens hat, wird in diese Arbeit beispielhaft durchgeführt.

1.2 Zielsetzung

Diese Arbeit verfolgt folgende Ziele:

1. Die *Definition einer SOA* soll dargestellt werden.
2. Ein *Qualitätsmodell zur Architekturbewertung* soll hergeleitet werden.
3. Das *Nutzenpotential* der Softwarearchitektur *SOA* soll anhand des aufgestellten Qualitätsmodells *bewertet werden*.
4. In einer Wirtschaftlichkeitsbewertung soll *beispielhaft die Wirtschaftlichkeit einer SOA* bewertet werden.

Potthof stellt in einem Literaturreview über empirische Studien des Nutzens der IT fest, dass die Frage nach dem Nutzen der IT nicht nur berechtigt, sondern immer wieder gestellt und untersucht werden muss.[61] Erwartungsgemäß liefert die empirische Forschung noch keine überzeugenden Ergebnisse.[62] Ebenso kommen Brynjolfsson und Yang zu dem Ergebnis, dass eine einheitliche Nutzenbewertung schwer möglich ist.[63] Irani und Love stellen genauso fest, dass es nicht möglich ist, einen generischen Bewertungsprozess, der für alle IT-Anwendungen verwendbar ist, aufzustellen.[64] Sie fordern stattdessen Bewertungsprozesse, die auf spezifische Typen von Anwendungen ausgerichtet

---

[60] Artikel über SOA, die (teilweise nebenbei oder spekulativ) über die Wirtschaftlichkeit im Praxiseinsatz berichten, stellen aktuell Einzelbetrachtungen in Form von Fallstudien dar. vgl. z. B. Schlamann /Service Orientation/, Hagen /Credit Suisse/, Berensmann /Postbank/ oder Bath, Herr /Post/.

[61] Potthof /Empirische Studien/ S. 63.

[62] Vgl. Potthof /Empirische Studien/ S. 63; Alshawi, Irani, Baldwin /Benchmarking/ S. 418; Irani, Love /Propagation/ S. 163.

[63] Vgl. Brynjolfsson, Yang /Productivity/ S. 179-182 & S. 207-208.

[64] Vgl. zum folgenden Absatz Irani, Love /Systems Evaluation/ S. 184-185.

sind.[65] Aus diesem Grund wird ein Qualitätsmodell zur Architekturbewertung hergeleitet.

Des Weiteren wird in dieser Arbeit ein Bewertungsrahmen für eine spezielle Ausprägung einer integrierten organisatorischen IT-Infrastruktur aus Sicht eines Unternehmens aufgestellt: Eine Wirtschaftlichkeitsanalyse von Unternehmens-IT-Systemen, die nach Prinzipien einer SOA aufgebaut ist. Wirtschaftlichkeitsanalyse ist ein weiterer Begriff, der eine Vielzahl Methoden umfasst, mit denen die Wirtschaftlichkeit in Form von Kosten und Nutzen geschätzt bzw. berechnet werden kann.[66] Es ist nicht möglich, eine generische Wirtschaftlichkeitsanalyse für alle möglichen Bewertungsfälle aufzustellen.[67] Folglich wird sich eine allgemeingültige Aussage der Wirtschaftlichkeit einer SOA auf empirischer Basis als extrem schwierig erweisen. Aus diesem Grund werden auf theoretischer Grundlage Argumentationen aufgestellt, mittels denen eine Wirtschaftlichkeitsbewertung für den Einsatz einer SOA unterstützt werden kann.[68]

In dieser Arbeit wird eine *qualitative* Untersuchung durchgeführt.[69] Es werden Aussagen getroffen, dass eine Wirkung in eine Bewertungsrichtung – positiv oder negativ – entsteht. Es wird jedoch nicht festgelegt, in welchem Ausmaß diese Wirkung auftritt. Für Softwarearchitekturen kann nur eine qualitative Aussage getroffen werden.[70] Um eine quantitative Wirtschaftlichkeitsbewertung durchführen zu können, müssten alle Nutzeneffekte bekannt und monetär bewertbar sein.[71] Durch die qualitative Bewertung kann lediglich auf ein Wirtschaftlichkeitspotential geschlossen werden. Um von einem Wirtschaftlichkeitspotential auf eine spezifische Wirtschaftlichkeit schließen zu können, muss eine spezifische Situation betrachtet werden. Für eine spezifische Situation lässt sich ggf. eine quantitative Bewertung der Wirtschaftlichkeit erreichen.

---

[65] Vgl. zum folgenden Absatz Irani, Love /Systems Evaluation/ S. 184-185.

[66] Vgl. Sassone, Schaffer /CBA/ S. 3.

[67] Vgl. Sassone, Schaffer /CBA/ S. 3.

[68] Boer argumentiert beispielhaft, dass die Bewertung von Technologien nur im spezifischen Anwendungskontext vorgenommen werden kann, vgl. Boer /Valuation/ S. 72-75.

[69] Vgl. zum Unterschied zwischen quantitativen und qualitativen Bewertungsmethoden Posch, Birken, Gerdom /Basiswissen/ S. 175-180.

[70] Vgl. Kapitel 3.2.1.

[71] Vgl. Linß /Nutzeffekte/ S. 32. Die Einschränkung von Linß kann auch verallgemeinert werden: Die Bewertung muss anhand einer einheitlichen Skala und Einheit vorgenommen werden können. Eine monetäre Einheit mit linearer Skala bietet sich zur Wirtschaftlichkeitsbewertung selbstverständlich an.

1.3 Methode der Arbeit

Jede wissenschaftliche Theorie setzt bestimmte Annahmen von handelnden Individuen und Gruppen, von Kontext bzw. Struktur sowie von dem Verhältnis von Struktur und Handlung voraus. Diese Annahmen bestimmen wesentlich mit, welche Forschungsfragen gestellt und welche Methoden verwendet werden (können). Sie stehen tatsächlich im Zentrum der Thematisierung und Theoretisierung gesellschaftlicher Veränderung. Wie in dieser Arbeit vorgegangen wird, ist anhand der Vorgehensweise in Kapitel 1.3.2 dargelegt. Vorher wird in Kapitel 1.3.1 die dieser Forschungsarbeit zugrunde liegende Forschungsmethodologie vorgestellt. Um die Grenzen der Arbeit aufzuzeigen, wird eine Abgrenzung der Arbeit in Kapitel 1.3.3 vorgenommen.

1.3.1 Forschungsmethodologie

Die dieser Forschung zugrunde liegende Forschungsmethodologie ist die des ‚kritischen Realismus'.[72] Mingers[73] stellt die Eignung des kritischen Realismus für die Forschung im Bereich der Wirtschaftsinformatik dar. Demnach ist der kritische Realismus eine konsistente, zusammenhängende und unterstützende Forschungsphilosophie für die Wirtschaftsinformatik. [74]

Der kritische Realismus kann kurz als kreativer Prozess dargestellt werden, in dem Beispiele und Gegenbeispiele oder Argumente und Gegenargumente gefunden und gegenüber gestellt werden, um zu einem Forschungsergebnis zu gelangen. Der kritische Realismus unterstützt eine pluralistische Sichtweise:[75] Die Vielfalt von Paradigmen und Methoden wird unterstützt, vorzugsweise gewünscht. Es ist akzeptiert, dass je nach For-

---

[72] Die Darstellungen in diesem Kapitel beruhen auf der Argumentation und Diskussion von Mingers, Monod und Klein zum Thema „Critical Realism and information systems". Diese Ausarbeitungen finden sich in: Mingers /Critical Realism/ und Mingers /Real-izing information systems/. Eine Auseinandersetzung mit diesem speziellen Themenbereich findet sich in der Spezialausgabe der „Information and Organization", Vol. 14 Nr. 2, 2004, zum Thema „Critical Realism". An der Diskussion in dieser Ausgabe beteiligen sich neben John Mingers Emmanuel Monod und Heinz K. Klein. Der kritische Realismus wurde im Zeitraum 1978 bis 1993 entwickelt.

[73] Mingers /Real-izing information systems/; Mingers /Critical Realism/.

[74] Vgl. Mingers /Real-izing information systems/ S. 88; Mingers /Critical Realism/ S. 374.

[75] Einem Pluralisten gegenüber können Imperialisten und Isolationisten unterschieden werden. Ein Imperialist unterstellt *ein dominantes Paradigma*, dem sich alle unterzuordnen haben. Isolationisten akzeptieren *verschiedene, jedoch schwer vergleichbare Paradigmen*, die für verschiedene Forschungsbereiche unterschiedlich gut geeignet sind. Dabei soll jedes Forschungsgebiet einem geeigneten Paradigma folgen. Vgl. Mingers /Critical Realism/ S. 373.

schungsfall bestimmte Paradigmen besser geeignet sind, die jeweiligen Forschungsfragen zu beantworten.

Forschungen im Feld der Wirtschaftsinformatik stützten sich klassisch auf positivistische Forschungsphilosophien (eine der bekanntesten positivistischen Methoden ist die empirische Forschung).[76] Positivistischen Forschungsphilosophien stehen einige Kritiken gegenüber:

- Empirische Zusammenhänge sind ein ‚psychologischer Glaube',[77]
- Beobachtung und Auffassung sind problematisch,[78]
- Universal gültige Gesetzte können nicht aus einer Menge von Beobachtungen abgeleitet werden,[79] und die
- Verwendung von Falsifikationen ist ungenau.[80]

Dem Positivismus gegenüber steht der Interpretivismus. Dem Interpretivismus nach ist der Zweck der Wissenschaft, hilfreiches Wissen durch das Verstehen sozial-politischer Interaktionen zu produzieren, und im Gegensatz zum Positivismus nicht, „die Welt verstehen [zu] wollen."[81] Auch der Interpretivismus sieht sich Kritiken gegenüber:

---

[76] Vgl. Mingers /Critical Realism/ S. 372.

[77] Die positivistische Forschung leitet aus einem Ereignis ‚*aus a folgt b*' ab, dass b immer aus a folgt. Alles was beobachtet werden kann, ist die konstante Folge von Ereignissen. Dass a b *immer* verursacht, ist der zugrundgelegte ‚psychologische Glaube'. Prinzipiell wird unterstellt, dass sich *allgemeingültige* Gesetze aus einer Reihe einzelner Beobachtungen ableiten lassen. Vgl. Mingers /Critical Realism/ S. 374.

[78] Der positivistischen Forschung liegt die Annahme zugrunde, dass Beobachtung und Auffassung unproblematisch sind. Vielfach konnte jedoch (theoretisch und praktisch) gezeigt werden, dass Beobachtungen *theorieabhängig* sind z. B. von der anerzogenen Sprachstruktur. (Vgl. hierzu Arbeiten von Piaget, Gregory, Hansen, Cicourel, und weiteren. Nach Mingers /Critical Realism/ S. 376.) Ein Beispiel hierfür ist, dass sich die Sonne um die Erde dreht.

[79] Die positivistische Wissenschaft beruht auf Induktion: Universal gültige Gesetzte können aus einer Menge von Beobachtungen abgeleitet werden. Dieser induktive Ansatz wird angezweifelt, weil der zugrunde liegende Kausalismus, dass etwas, was einmal beobachtet wurde, immer wieder beobachtet werden kann, eine Annahme ist. Vgl. Mingers /Critical Realism/ S. 374.

[80] Falsifikation wird in der positivistischen Wissenschaft zur Widerlegung einer Theorie verwendet. Das Problem einer Falsifikation ist allerdings, dass nicht nur die eigentliche Theorie widerlegt werden kann. Ein fehlgeschlagenes Experiment kann durchaus ‚nur' unterstützende Theorien falsifizieren. Auch kann eine Falsifikation den Aufbau eines Experiments falsifizieren – unabhängig von der zugrunde gelegten Theorie. Zuletzt beruht Falsifikation selber auf Induktion, die auch kritisiert wird. Vgl. Mingers /Critical Realism/ S. 375-377.

[81] Mingers /Critical Realism/ S. 377.

- Wissenschaftliche Theorien sind von der menschlichen Wahrnehmung, Konzeptualisierung und Bewertung abhängig,[82]
- Innovationen werden ignoriert[83]

Kritischer Realismus (Critical Realism)

Für den Positivisten existiert nur das, was erfahren werden kann. Für den Interpretivisten sind Grenzen des eigenen Wissens auch die Grenze des Existentiellen. In der Auffassung eines kritischen Realisten dagegen existiert die Welt auch unabhängig von der Menschheit,[84] und Etwas, was Auswirkungen auf die Welt hat, existiert – selbst dann, wenn es nicht wahrgenommen oder gemessen werden kann.

Das grundlegende Interesse eines Realisten ist das Erklären.[85] Die Methode der Realisten kann als eine „herleitende Methode" charakterisiert werden. Der Forscher betrachtet beobachtete unerklärte Phänomene und unterstellt hypothetische Mechanismen, die – wenn diese existieren würden – das bedingen würden, was zu erklären ist. Dass solche Mechanismen wirklich existieren wird im kritischen Realismus jedoch nicht als sicher angenommen. Es wird hingenommen, dass Wissen fehlbar (dies impliziert auch unvollständiges Wissen) sein kann.[86] Ein kritischer Realist versucht Wissen zu vervollständigen, indem er alternative Erklärungen durch Herleitung potentieller Effekte eliminiert. Ein kritischer Realist will unter ‚das Oberflächliche' sehen, um zu verstehen, und um zu erklären, warum Dinge so sind, wie sie sind. Er stellt Hypothesen auf, um die Strukturen und Mechanismen, die beobachtbare Ereignisse ummanteln, zu erklären.[87]

Ziele des kritischen Realismus sind:[88]

---

[82] Kritisiert wird insbesondere, dass Interpretivisten unterstellen, dass ohne eine Wahrnehmung, Konzeptualisierung und Bewertung durch den Menschen ein ‚Etwas' nicht existieren kann.

[83] Paradigmen entwickeln sich als Konsens innerhalb sozialer Gemeinschaften aus Wissenschaftlern durch praktische Mechanismen wie Veröffentlichungen oder soziale Verbünde, Wissenschaftler treten durch die Akzeptanz der verwendeten Forschungspraxis der Forschungsrichtung hinzu. Hier werden Innovationen ohne überhaupt beachtet zu werden ignoriert, es sei denn, viele solcher ‚Anomalien' treten auf. Vgl. Mingers /Critical Realism/ S. 378.

[84] Vgl. Mingers /Critical Realism/ S. 383.

[85] Vgl. Mingers /Critical Realism/ S. 385; Mingers /Real-izing information systems/ S. 94-95.

[86] Vgl. Mingers /Critical Realism/ S. 385.

[87] Vgl. Mingers /Critical Realism/ S. 398.

[88] Vgl. Mingers /Critical Realism/ S. 380.

1. *Realismus*

   Akzeptanz der Existenz einer Welt, die ohne die Menschheit existiert. Gleichzeitig wird wahrgenommen, dass Wissen bzw. Erfahrung unvermeidlich durch die kognitiven Fähigkeiten und Erfahrungen des Menschen geprägt sind.[89] Wissenschaft ist nicht nur das Aufnehmen konstanter Beziehungen beobachtbarer Ereignisse, sondern handelt (1) von existierenden Objekten und Strukturen[90] (unabhängig, ob diese beobachtet werden oder ob Beobachter anwesend sind) und (2) von Ereignissen, die beobachtet oder erzeugt werden.

2. *Kritische* Grundeinstellung

   In sozialwissenschaftlichen Domänen soll eine kritische positivistische Position, in naturwissenschaftlichen Domänen eine kritische interpretivistische Position eingenommen werden können.[91] Folglich akzeptiert diese kritische Grundeinstellung die grundlegende (und bevorzugte) Forschungsphilosopie der jeweiligen anderen Domäne. Für die *Wirtschaftsinformatik* ist dieser Punkt von besonderer Relevanz: IT-Systeme sind sowohl in der sozialwissenschaftlichen als auch der naturwissenschaftlichen Domäne angesiedelt: Sie handelt (1) von technischen Systemen, die klaren, messbaren Regeln unterliegen. IT-Systeme sind (2) in sozialen Organisationen eingebettet.[92]

---

[89] „Re-establish a realist view of being in ontological domain while accepting the conditions of the epistemological domain." Mingers /Critical Realism/ S. 380.

[90] Bezüglich Objekten und Strukturen beachtet der kritische Realist, dass Menschen Ereignisse und Gegenstände falsch oder unterschiedlich auffassen können und dass Ereignisse und Gegenstände existieren können, unabhängig, ob diese beobachtet wurden oder Beobachter anwesend sind. Voraussetzungen für Wissen entspringt in der Auffassung eines kritischen Realisten nicht der Vorstellung, sondern durch die Struktur der Realität. Wenn ‚Etwas' „Einfluss auf die Welt hat, impliziert dies Existenz." Vgl. Mingers /Critical Realism/ S. 381-383, Zitat S. 383. Der kritische Realismus baut bezüglich dieser Sichtweise ein *Schichtenmodell* auf. Dieses besteht aus „the Real, the Actual and the Empirical." (Mingers /Critical Realism/ S. 384) Elemente des ‚Real' sind alle Arten von Mechanismen und Gegenständen. ‚The Actual' beinhaltet Ereignisse, die auftreten können, oder auch nicht. ‚The Empirical' sind Ereignisse, die wahrgenommen werden können und die tatsächlich beobachtet werden. Das, was empirisch nachgewiesen werden kann (‚the Empirical'), ist nur ein Ausschnitt aus ‚the Actual' und dies wiederum ein Ausschnitt aus ‚the Real'. Vgl. Mingers /Critical Realism/ S. 384; Mingers /Real-izing information systems/ S. 93.

[91] Naturwissenschaftler vertreten i. d. R. die Positionen von Positivisten, Sozialwissenschaftler dagegen i. d. R. die von Interpretivisten (vgl. Mingers /Critical Realism/ S. 379). Realisten akzeptieren diesen Unterschied zwischen Natur- und Sozialwissenschaft und vertreten die Meinung, dass sich die Positionen vereinbaren lassen. In Ergänzung dazu wird auch argumentiert, dass sich ‚*kritisch*' auf Habermas' „Critical Social Theory" bezieht. Vgl. Mingers /Critical Realism/ S. 381.

[92] Vgl. Mingers /Critical Realism/ S. 379.

Kritikpunkte am kritischen Realismus sind:[93]

1. Gültigkeit des Schichtenmodells,[94]
2. Methodische Schwächen,[95]
3. Natur der Wahrheit,[96]
4. Übertragbarkeit auf Sozialwissenschaften,[97]
5. Anspruch ‚kritisch' zu sein ist ein politischer Anspruch.[98]

Trotz dieser Kritikpunkte kann der kritische Realismus als anerkannt und verwendbar angesehen werden, weil diese Forschungsphilosophie seit einigen Jahrzehnten diskutiert und vor allem auch angewendet wird.[99]

1.3.2 Vorgehensweise

Literatur über SOA findet sich seit wenigen Jahren vermehrt.[100] Bei genauer Untersuchung der Beiträge kann festgestellt werden, dass Artikel existieren, dir nicht durchgängig wissenschaftliche Anforderungen erfüllen.[101] An wenigen Stellen der vorliegenden

---

[93] Vgl. Mingers /Critical Realism/ S. 388-393.

[94] In dem Schichtenmodell (vgl. Fußnote 90) werden eine nicht-menschliche und nicht-empirische Welt vereint. Dieser Versuch der Vereinigung der ‚menschlichen' und ‚empirischen' Welt ist für Forscher aus den jeweiligen Gebieten nicht akzeptabel. Vgl. Mingers /Critical Realism/ S. 388-393.

[95] Durch die Akzeptanz der menschlichen Wissensproduktion (Interpretivismus) und der Theorieabhängigkeit einer Beschreibung (Positivismus) ist eine erste Beschreibung des Forschungsgegenstandes bereits durch unterstellte Konzepte beeinflusst. Die Eliminierung und Identifikation von Mechanismen ist ein kreativer Prozess und kann zu untestbaren, unerwarteten und unvollständigen Ergebnissen führen. Vgl. Mingers /Critical Realism/ S. 389-390.

[96] Durch die Akzeptanz einer erkenntnistheoretischen Relativität kann Wahrheit nicht in allerletzter Instanz vorhergesagt werden. Vgl. Mingers /Critical Realism/ S. 390-390.

[97] Die Übertragbarkeit auf Sozialwissenschaften wird in Frage gestellt. Vgl. Mingers /Critical Realism/ S. 391-391.

[98] Kritisch wird hauptsächlich die Methodologie gehandhabt, nicht erzeugte substantielle Theorien – die für Forscher allerdings von eigentlichem Interesse sind. Dies führt dazu, dass keine kritische Forschung betrieben wird, sondern eine kritische Philosophie der verwendeten Forschungsmethodik zugrunde gelegt wird. Vgl. Mingers /Critical Realism/ S. 392-393.

[99] Vgl. Mingers /Critical Realism/ S. 380.

[100] Z. B. erschien im Mai 2004 ein Cutter IT Journal (Vol. 17, Nr. 5) zum Thema SOA. Zudem werden immer mehr Bücher wie Erl /Service-Oriented Architecture/, Banke, Krafzig, Slama /Enterprise SOA/ und Woods /Enterprise Services Architecture/ zum Thema SOA veröffentlicht.

[101] Beispielsweise die auffindbare Literatur von Beratungs- und Marktforschungsunternehmen wie Gartner (z. B. Natis), META Group (z. B. Scholler), Unilog Avinci (z. B. Kuhn /Trends/) und auf das Thema ‚Integration' spezialisierte Zeitschriften, wie z. B. das ‚Business Integration Journal', in dem allerdings auch bekannte und renommierte Autoren wie David S. Linthicum publizieren. An dieser

Arbeit wurde gleichwohl auf diese Beiträge zurückgegriffen, weil sie qualitativ gut sind, wertvolle Informationen beinhalten und einzelne Aussagen der vorliegenden Arbeit (teilweise empirisch oder durch Fallstudien) bestätigen.[102]

Schon 1978 stellte Barry Boehm fest, dass die eine alles umfassende Metrik um Softwarequalität bewerten zu können schwer zu finden ist, und die Verwendung einer Metrik für alle Software-Bewertungsfälle sogar kontraproduktiv sein kann.[103] Momentan hat sich an dieser Meinung nichts geändert – sie wird weiterhin vertreten und manifestiert sich in einer Sammlung unterschiedlicher Qualitätsmodelle, die zur Bewertung von Software herangezogen werden.[104]

Die Untersuchung in dieser Arbeit folgt einem wertbezogenen Ansatz.[105] Ein wertbezogener Ansatz folgt der Ansicht, dass Qualität eines Produktes zustande kommt, weil ein Nutzen zu einem bestimmten Preis erreicht wird. Folglich werden zur Wirtschaftlichkeitsbewertung einer SOA ein Nutzenpotential sowie ein Kostenpotential erarbeitet und diese bezüglich wirtschaftlicher Kriterien gegenübergestellt. Bei der Identifizierung von Kosten werden nur diese in die Bewertung einbezogen, die sich durch die Struktur und die verfolgten Gestaltungsziele einer SOA ergeben. Kostenfaktoren werden erhoben, indem aus Fallstudien und Praxisberichten spezifische Kosten identifiziert werden, die explizit durch den Einsatz einer SOA entstanden sind. Des Weiteren werden Kostenfaktoren durch Argumentation identifiziert, die darlegen, in welchen Bereichen der Einsatz von SOA Kosten verursacht.

Stelle sei darauf hingewiesen, dass diese Beiträge auch nicht das Ziel verfolgen wissenschaftlichen Ansprüchen genügen zu wollen, sondern vielmehr aus der Praxis und für die Praxis Wissen vermitteln wollen.

[102] Einige Beispiele der verwendeten und im klassischen Sinn als graue Literatur einzustufende Beiträge sind: Adam, McKendrick /Everything in Between/. (Adam und McKendrick führten eine Umfrage unter fast 1000 Managern und Entwicklern zum Thema SOA und WebServices durch. Die Ergebnisse sind für diese Arbeit zur Darstellung der Problematik des Verständnisses und der Wahrnehmung von SOA in der Praxis hilfreich.), Artikel der META Group und von Gartner (die META Group stellte in einigen Fallstudien dar, welche Nutzenpotentiale sich in der Praxis tatsächlich ergaben.), CapGemini /IT-Trends 2005/ (Dieser Beitrag ist eine Marktstudie, dessen erhobenes Datenmaterial zur Rechtfertigung der Signifikanz einzelner Argumente bzw. geschäftlicher Anforderungen an Unternehmens-IT beiträgt) und Huizen /SOA/ (van Huizen berichtet im Business Integration Journal über seine Projekterfahrung bei der Implementierung einer SOA.)

[103] Vgl. Boehm u. a. /Characteristics/ S. (3-1)-(3-3).

[104] Vgl. beispielsweise IEEE /Architectual Description/ S. 7.

[105] Vgl. zu einer Zusammenfassung unterschiedlichen Ansätzen der Qualitätsbewertung Mellis /Lehre/ S. 386-389.

Um Ansatzpunkte für eine Wirtschaftlichkeitsanalyse zu finden und um potentielle Muster in Bezug auf die Wirtschaftlichkeit einer SOA aufzudecken, wurden Diplomarbeiten aufgesetzt, die in Fallstudien die Wirtschaftlichkeit einer SOA untersuchten. In diesen Arbeiten wurde der technisch-fachliche Nutzen einer SOA evaluiert, eine Risikoanalyse für SOA-Projekte durchgeführt und die Eignung einer SOA für eine Branche untersucht. Die Untersuchungen lieferten wertvolle Hinweise für diese Forschungsarbeit. Über die Diplomarbeiten hinaus wird in dieser Arbeit ein Qualitätsmodell speziell für Softwarearchitekturen hergeleitet, welches auch geschäftliche Kriterien beinhaltet. Des Weiteren wird in dieser Arbeit eine allgemeine qualitative Bewertung angestrebt, wohingegen die Diplomarbeiten Fallstudien mit verschiedenen Schwerpunkten darstellen. Diese Vorgehensweise stimmt mit Vorschlägen kritischer Realisten überein, um substantielle Forschungen auf eine gute Ausgangsbasis stellen zu können.[106]

Im zweiten Kapitel wird eine Arbeitsdefinition SOA aufgestellt. Hier wird zunächst eine Definition für Softwarearchitektur dargelegt und anschließend die Struktur von SOA erläutert. Im dritten Kapitel wird das Nutzenpotential von SOA bewertet. Dazu wird zunächst ein Qualitätsmodell für Softwarearchitekturen gebildet. Anhand dieses Qualitätsmodells wird das Nutzenpotential von SOA qualitativ bewertet. Das vierte Kapitel bewertet das Kostenpotential, das SOA aufweist. Im fünften Kapitel wird eine beispielhafte Wirtschaftlichkeitsanalyse für SOA vorgenommen. Im sechsten Kapitel werden empirische Ergebnisse vorgestellt und diskutiert. Im siebten und letzten Kapitel wird ein Resümee gezogen, die Arbeit kritisch gewürdigt und beschrieben, für welche weiteren Verwendungsmöglichkeiten das aufgestellte Wirtschaftlichkeitsmodell eingesetzt werden kann. Der Aufbau der Kapitel zwei bis fünf ist schematisch in Abb. 1-1 dargestellt.

1.3.3 Abgrenzung der Arbeit

SOA können mit unterschiedlichen Technologien realisiert werden.[107] Aus diesem Grund sind WebServices eine,[108] aber nicht die einzige Alternative, die als technische

---

[106] Vgl. Mingers /Critical Realism/ S. 395-396.

[107] Vgl. Gold u. a. /Understanding/ S. 72.

[108] Vgl. Linthicum /SOA/ S. 15; Dostal, Jeckle /Service-orientierte Architektur/ S. 54; Adam, McKendrick /Everything in Between/ S. 19; Carlson, Tyomkin /Good Design/ S. 14; Erl /Service-Oriented Architecture/ S. 51; Natis /SOA Scenario/ S. 1.

Grundlage zur Erstellung einer SOA genutzt werden kann.[109] SOA ist demnach eine Softwarearchitektur, die mit verschiedenen Technologien umgesetzt werden kann.[110] Z. B. kann die Kommunikationsinfrastruktur für Services ein einfaches verwaltetes Netzwerk, eine nachrichtenorientierte Middleware oder eine umfangreiche Integrationsplattform sein.[111]

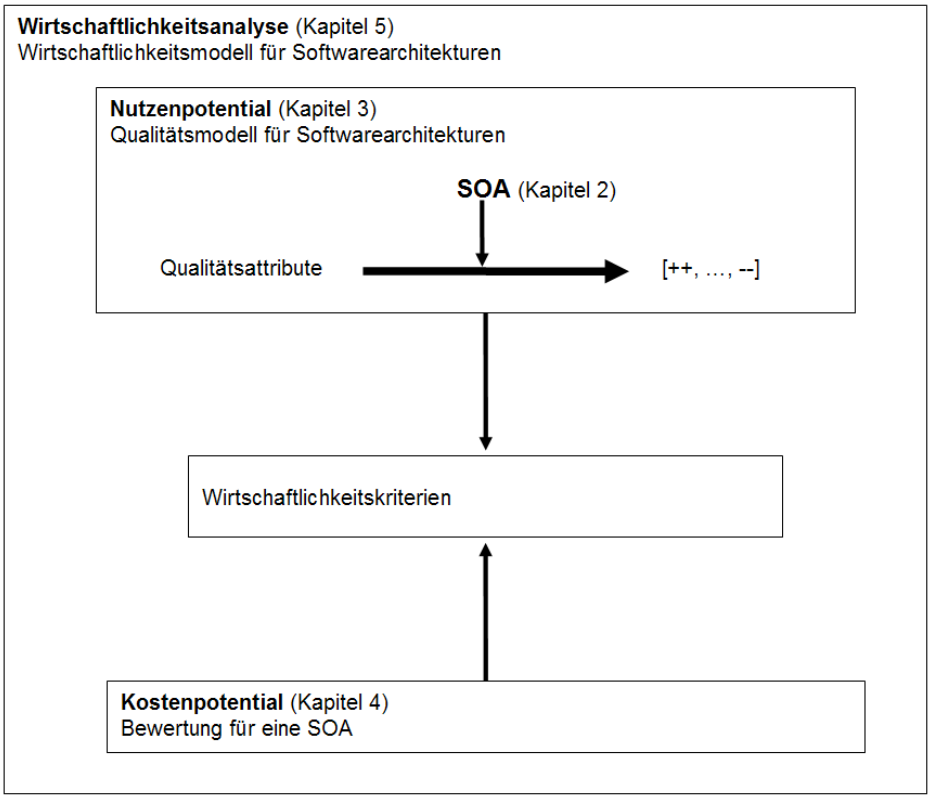

Abb. 1-1: Aufbau der Arbeit

In Einklang mit der Definition einer SOA (vgl. Kapitel 2.2.2) sind vielfältige Möglichkeiten der Gestaltung einer SOA mittels verfügbaren Technologien und Produkten möglich. Schon die Aufzählung der potentiell einsetzbaren Technologien und Produkte würde mehrere Seiten umfassen.[112] Eine Aufzählung dieser ist für diese Arbeit nicht

---

[109] Vgl. Gold u. a. /Understanding/ S. 72; Natis /SOA/ S. 23; Carlson, Tyomkin /Good Design/ S. 14.

[110] Vgl. Erl /Service-Oriented Architecture/ S. 51; Gold u. a. /Understanding/ S. 72.

[111] Vg. Dostal, Jeckle /Service-orientierte Architektur/ S. 54, die Autoren stellen dies beispielsweise für WebServices fest.

[112] Z. B. müssten Programmiersprachen, Datenformate, Datenbanken, Middlewarekonzepte und –produkte, Formate zum Datentransport, und vieles mehr beachtet werden.

nutzenrelevant, weshalb darauf verzichtet werden kann.[113] Für die Bewertung eine Softwarearchitektur bei Beachtung spezifischer technischer Alternativen müssen viele Eigenschaften der entsprechenden Technik berücksichtigt werden. Weil alle Alternativen technischer Umsetzung in eine Bewertung einfließen müssten, um eine Gesamtbewertung präsentieren zu können, würde der Umfang dieser Arbeit um ein Vielfaches überschritten werden. Informationen müssten zu jeder technischen Kombination zur Umsetzung einer SOA, zu jeder Kombination dieser technischen Umsetzungsmöglichkeiten und zu den entsprechenden Eigenschaften dieser Kombinationen erhoben werden. Anschließend müssten alle Kombinationen und ihre Eigenschaften bewertet werden. Solche Informationen würden nicht nur den Rahmen der Arbeit sprengen, sondern auch zweifelhaften Aussagewert besitzen, weil ein entsprechender Einsatz von Technologien nicht nur von einer solchen Bewertung abhängt, sondern auch anderen Entscheidungsfaktoren innerhalb eines Unternehmens unterliegt. Darunter fallen z. B. bestehende Geschäftsbeziehungen, Erfahrungen der Entwickler, vorhandenes Know-How und Ähnliches.

Eine Bewertung aller Möglichkeiten zur technischen Gestaltung einer SOA ist folglich auf allgemeiner Ebene nicht möglich. Doch auch schon für einen konkreten Fall müsste eine Vorauswahl anhand von festgelegten Kriterien getroffen werden, damit eine Bewertung weniger Alternativen durchgeführt werden kann. Die Aussagefähigkeit einer solchen Betrachtung wäre sehr eingeschränkt und würde zur Lösung des Praxisproblems wenig beitragen. Aus diesem Grund abstrahiert diese Arbeit vollständig von Technologien und Produkten.

1.4 Begriffsklärung

Im Folgenden werden einige in dieser Arbeit verwendeten Begriffe näher erläutert und teilweise gegenüber anderen Begriffen abgegrenzt.

Der Begriff ‚**Service**' ist der Name einer Softwarekomponente einer SOA. Ein Service erbringt eine abgegrenzte ‚**Dienstleistung**'. Dienstleistung beschreibt die Funktionen,

---

[113] Der interessierte Leser findet Informationen in der vielfältig vorhandene Literatur zu entsprechenden Themen und Techniken: Beispielsweise Gruhn, Thiel /Komponentenmodelle/, Hagen /Credit Suisse/, Karch u. a. /SAP/, Bath, Herr /Post/, Berensmann /Postbank/, Erl /Service-Oriented Architecture/, Hauser, Löwer /Web Services/, Ließmann, Wetzke /Integrationsinfrastruktur/, Moro, Lehner

mittels derer ein Service Anforderungen erfüllt. Folglich erweitert diese Arbeit den deutschen Begriff ‚Service'[114] (z. B. aus dem Tennis) und nutzt ihn als Bezeichnung für Softwarekomponenten einer SOA.[115]

Ein **Geschäftsprozess** stellt eine Menge von Aktivitäten dar, die in Sequenzen ausgeführt werden, um ein Ziel zu erreichen.[116] Ein Geschäftsprozess wird i. d. R. innerhalb eines Unternehmens durchgeführt. Ein Geschäftsprozess kann jedoch auch auf unternehmensübergreifender Ebene definiert werden.

Ein **Qualitätsattribut** ist eine Eigenschaft, die die Qualität eines Gegenstandes beeinflusst.[117] Qualitätsattribute können hierarchisiert werden.[118] Dadurch ergeben sich die Ebenen der Qualitätsattribute und der **Unterqualitätsattribute**.[119] Unterqualitätsattribute sind eine verfeinerte Ebene eines Qualitätsattributs. Ein **Qualitätsmodell** fasst Quali-

---

/Reengineering/, Österle /Integration/, Pasley /BPEL/, Schmietendorf, Dimitrov, Dumke /JavaBeans/, Stiemerling /Web-Services/ oder Zahavi /Integration/.

[114] Vgl. Wermke, Kunkel-Razum, Scholze-Stubenrecht /Duden/ S. 888.

[115] An dieser Stelle sei darauf hingewiesen, dass dies eine deutsche Bezeichnung ist. Man darf ‚Service' dieser Arbeit nicht als englischen Begriff auffassen und ihn ins Deutsche (z. B. als ‚Dienstleistung') übersetzten. Eine englische Bezeichnung des Begriffs würde z. B. ‚service-component' lauten.

[116] Vgl. Lublinsky, Tyomkin /Dissecting SOA/ S. 54.

[117] IEEE /Glossary/ S. 60.

[118] Die Bildung von Hierarchien hängt vom Gegenstand der Bewertung. Bass, Clements und Kazman weisen darauf hin, dass Qualitätsattribute in verschiedenen Zusammenhängen durchaus unterschiedlich hierarchisiert werden können. Vgl. Bass, Clements, Kazman /Software Architecture/ S. 281.

[119] Das IEEE nennt die Attribute der aggregierenden Ebene ‚Quality Factors' (Qualitätsfaktoren) und die der detallierten Ebene ‚Quality Attributes' (Qualitätsattribute). Vgl. IEEE /Glossary/ S. 60.

tätsattribute und ihre Unterqualitätsattribute zusammen. Mittels eines Qualitätsmodells können verschiedene Typen einer Bewertungsklasse einheitlich bewertet werden. In dieser Arbeit wird z. B. ein Qualitätsmodell zur Bewertung von Softwarearchitekturen aufgestellt.

Unternehmens-IT bezeichnet hier ein IT-System, das im betrieblichen Kontext zur Erfüllung bzw. Unterstützung von Aufgaben eingesetzt wird. Andere Autoren verwenden andere im jeweiligen Zusammenhang synonyme Begriffe: Gronau z. B. Informationssysteme[120]; Griffiths und Finlay z. B. IS (Informationssysteme).[121]

[120] Vgl. Gronau /Wandlungsfähige Informationssystemarchitekturen/ S. 45.

[121] Vgl. Griffiths, Finlay /Competetive Advantage/ S. 50–51.

2. SOA als Konzept einer Softwarearchitektur

SOA kann als Weiterentwicklung der Komponentenorientierung verstanden werden, weil die Idee einer SOA aus den bereits existierenden Konzepten zur Komponenten- und Objektorientierung, die weiter ausgearbeitet und verfeinert wurden, entstand.[122]

Grundlegend ist eine SOA ein Architekturkonzept für eine unternehmensweite Architektur von Softwaresystemen, also für die gesamte Unternehmens-IT; es kann auch auf ein einzelnes Softwaresystem angewendet werden. Das Architekturkonzept, welchem SOA zugrunde liegt, ist das einer Softwarearchitektur. Der Begriff ‚Softwarearchitektur' stellt somit die Basis für ein Verständnis von SOA und für das Verständnis der Trageweite von SOA dar. Aus diesem Grund wird im Folgenden zuerst der Begriff Softwarearchitektur erläutert. Wie der Begriff SOA im Kontext von Softwarearchitektur definiert wird, wird in Kapitel 2.2 vorgestellt. Im Kapitel 2.3 wird schließlich eine Abgrenzung serviceorientierter Architekturen zu Konzepten wie Enterprise Application Integration und ereignisgesteuerten Architekturen gezogen.

2.1 Softwarearchitektur

Architekturen stellen Gestaltungsmodelle dar, die Bausteine einer Lösung in einen Zusammenhang bringen.[123] Dieses Verständnis lässt sich auch auf Software übertragen, weil auch Software in Bausteine unterteilt und in Gestaltungsmodellen[124] dargestellt werden kann. Eine Architektur für Software wird als Softwarearchitektur bezeichnet. Softwarearchitektur kann mit der Architektur einer Stadt verglichen werden.[125] So, wie eine Stadt aus Straßen, Wohnblöcken, Fußgängerzonen und Stadtwerken besteht, kann auch Software in ihre Komponenten – wie Netzwerke, Datenbanken, Benutzeroberflächen und Transportmechanismen – unterschieden werden. So, wie Komponenten einer Stadt unterschiedlich strukturiert werden und unterschiedlich miteinander interagieren,

---

[122] Vgl. Dostal, Jeckle /Service-orientierte Architektur/ S. 53. Daher stellt man sich die Frage, inwieweit eine SOA nur „Old wine in new bottles" oder „new vintage" darstellt, vgl. Allen /Statement/ S. 2.

[123] Vgl. Sinz /Architektur/ S. 875; Strunz /Anwendungsarchitektur/ S. 35; Inmon, Zachman, Geiger /Data/ S. 30-33.

[124] Beispielsweise seien hier die vielfältigen Möglichkeiten der UML, des ERM oder der Use-Case-Modellierung genannt.

[125] Vgl. zu diesem Absatz Burke /Enterprise Architecture/ S. 1-2.

können auch die Komponenten einer Software unterschiedlich strukturiert werden und unterschiedlich miteinander interagieren.

Für die Betrachtung von Softwarearchitekturen können unterschiedliche Betrachtungsebenen angewendet werden, die zu verschiedenen Sichtweisen führen, wie die Grenzen des betrachteten Softwaresystems gegenüber anderen Systemen gezogen werden. Diese unterschiedliche Betrachtungsebene führt dazu, dass, wie auch oben schon angesprochen, SOA zum einen im Rahmen der Konstruktion einzelner Anwendungssysteme eines Unternehmens eingesetzt werden; zum anderen können sie jedoch auch als unternehmensweites und anwendungsübergreifendes Konzept zur Gestaltung der betrieblichen Datenverarbeitung angewendet werden.

In dieser Arbeit soll die Bedeutung des Architekturkonzeptes für Unternehmen betrachtet werden. Der Begriff ‚Unternehmensarchitektur' wird wirtschaftswissenschaftlich als ein konzeptioneller Rahmen verstanden, der beschreibt, wie ein Unternehmen aufgebaut ist, und wie seine Komponenten und die Beziehungen dieser Komponenten untereinander gestaltet sind.[126] Übertragen auf die Unternehmens-IT bedeutet dies, dass eine unternehmensweite Softwarearchitektur die Architektur aller unternehmensweit eingesetzten Softwaresysteme abbildet und darstellt, wie diese aufgebaut sind, wie ihre Komponenten und wie die Beziehungen der Komponenten untereinander gestaltet sind.

Betrachtungsebenen einer Softwarearchitektur können in eine physische und eine logische Betrachtungsebene unterschieden werden.[127] Zahavi trennt diese beiden Ebenen voneinander und unterstreicht die Wichtigkeit, die Grenzen dieser Betrachtungsebenen nicht zu verwischen, um zu vermeiden, dass Unklarheiten, welche durch eine unzureichende Trennung der Ebenen entstehen, ein Projekt in Verzug bringen.[128] Allerdings wird die integrierte Betrachtung beider Ebenen notwendig, weil Konzepte zu Softwarearchitekturen heute Gestaltungsprinzipien beinhalten, die beide Betrachtungsebenen betreffen. Für den Fall SOA wird beispielsweise angenommen, dass Softwarekompo-

---

[126] Rood /Enterprise Architecture/ S. 106; Kieser, Walgenbach /Organisation/ S. 16-23. Kieser und Walgenbach gehen in ihren Ausführungen auf formale Organisationsstrukturen ein und stellen dar, welchen Regeln sich Mitarbeiter eines Unternehmens bezüglich der Organisationsstruktur unterwerfen. Die Beschreibung stellt implizit dar, dass Komponenten der Organisation existieren und miteinander interagieren.

[127] Vgl. Zahavi /Integration/ S. 46-48.

[128] Vgl. Zahavi /Integration/ S. 47.

nenten lose gekoppelt aufgebaut sein sollten[129] und dadurch physisch an unterschiedlichen Stellen ausgeführt werden können. Hier wird somit sowohl eine physische als auch eine logische Betrachtungsebene integriert in einem Gestaltungskonzept betrachtet.

Über die Betrachtungsebenen hinaus kann eine Softwarearchitektur aus verschiedenen Perspektiven beschrieben werden. Dabei kann z. B. der Aufbau des Systems, das dynamische Verhalten der Systemkomponenten oder die Vorgehensweise zur Entwicklung des Softwaresystems im Mittelpunkt der Betrachtung stehen.[130] Aus diesen Gründen existieren verschiedene Auffassungen darüber, was unter einer Softwarearchitektur zu verstehen ist.[131] Viele Definitionen teilen das gemeinsame Verständnis, dass eine Softwarearchitektur aus Elementen bzw. Komponenten[132] besteht und deren Beziehung zueinander beschreibt. In dieser Arbeit wird der Definition von Bass, Clements und Kazman gefolgt, die definiert, dass ‚**Softwarearchitektur die Struktur (oder die Strukturen) eines Softwaresystems ist, welche (a) *Softwarekomponenten*, (b) eine Beschreibung der *extern sichtbaren Eigenschaften* der Softwarekomponenten und (c) eine Beschreibung der *Beziehungen der Softwarekomponenten* zueinander beinhaltet.**'[133] (Vgl. Abb. 2-1.)

2.2 Die Softwarearchitektur SOA

In der Wissenschaft und Praxis wird seit einiger Zeit intensiv über das Konzept serviceorientierter Architekturen debattiert. Es lässt sich allerdings feststellen, dass sich bisher kein einheitliches Verständnis des Begriffs SOA gebildet hat bzw. Anwendung fin-

---

[129] Vgl. hierzu 2.2.4.3.

[130] Vgl. Clements u. a. /Documenting Architecture/ S. 4.

[131] Belegt wird dies beispielsweise durch die Liste des SEI, die über 30 Definitionen für den Begriff der Softwarearchitektur auflistet, die aus jüngeren Veröffentlichungen, Veröffentlichungen größeren Einflusses und den vom SEI verwendeten Referenzen stammen. Vgl. SEI /Definitions/.

[132] IEEE definiert Komponente folgendermaßen: „One of the parts that make up a system. A component may be hardware or software and may be subdivided into other components. Note: The terms module, component, and unit are often used interchangeably or defined to be subelements of one another in different ways depending upon the context." (IEEE /Glossary/ S. 14.) In dieser Arbeit wird im Folgenden der Begriff **Softwarekomponente** genutzt. Dieser Begriff bezieht sich nur auf die Komponenten der *Software* einer Unternehmens-IT. Zwischen Modulen, Komponenten und Einheiten (bzw. ‚module, component and unit') wird in dieser Arbeit nicht unterschieden.

[133] „[The] software architecture of a program or computing system is the structure or structures of the system, which comprise software elements, the externally visible properties of those elements, and the relationships among them." (Vgl. Bass, Clements, Kazman /Software architecture/ S. 21.)

det.[134] Dieser Problematik widmet sich das direkt folgende Kapitel 2.2.1. Das Kapitel 2.2.2 stellt die verwendete Arbeitsdefinition von SOA vor. Anschließend wird in Kapitel 2.2.3 SOA aus der Perspektive des unternehmensweiten Kontexts vorgestellt. Um eine Softwarearchitektur erstellen zu können, müssen die jeweils angestrebte Struktur und die angestrebten Gestaltungsziele erläutert werden.[135] In Kapitel 2.2.4 wird die Struktur einer SOA erläutert und in Kapitel 2.2.5 werden die verfolgten Gestaltungsziele vorgestellt.

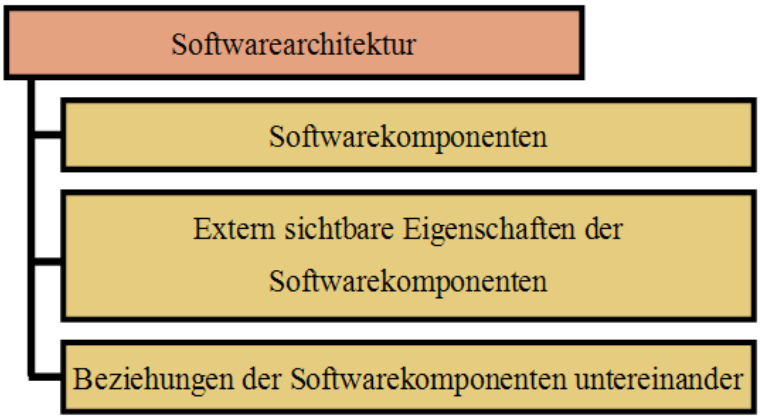

Abb. 2-1: Bestandteile der Definition ‚Softwarearchitektur'

2.2.1 Problematik eines uneinheitlichen Verständnisses der SOA

Trotz des hohen Stellenwertes, der dem Architekturkonzept SOA beigemessen wird,[136] hat sich bisher weder in der Praxis noch in der Forschung eine einheitliche Auffassung darüber gebildet, was unter dem Begriff SOA verstanden werden sollte.[137] Häufig ist unklar, worin die Besonderheiten des Architekturkonzeptes bestehen. Daher wird SOA mit der Kritik konfrontiert, nicht mehr als ‚alter Wein in neuen Schläuchen' zu sein.[138] Diese Kritik wird durch die Tatsache genährt, dass das Konzept der SOA in vielen As-

---

[134] Vgl. Aier, Schönherr /Flexibilisierung/ S. 39.

[135] Aier und Schönherr sprechen hier zusammenfassend von „Rahmenbedingungen". Vgl. Aier, Schönherr /Flexibilisierung/ S. 39.

[136] Vgl. Kapitel 1.1.

[137] Vgl. beispielsweise Adam, McKendrick /Everything in Between/ S. 32-33; Schlamann /Service Orientation/ S. 5.

[138] Vgl. Allen /Statement/ S. 2-4.

pekten eine Fortsetzung und Weiterentwicklung seit längerem bekannter und verbreiteter Prinzipien wie z. B. der Komponentenorientierung[139] darstellt.[140]

Obwohl der Begriff der SOA bereits seit Mitte der neunziger Jahre verwendet wird,[141] unterscheiden sich die vorgeschlagenen Definitionen noch immer stark hinsichtlich der Gewichtung einzelner Aspekte und des Detaillierungsgrades, mit dem sie erläutert werden. Die Besonderheiten des Architekturkonzeptes werden dabei häufig nicht ausreichend betont, um die genannte Kritik zu entkräften. Dies wird im Folgenden beispielhaft anhand der Definitionen der Gartner Group und des W3C dargestellt.

Gartner Group:

„SOA is a software architecture that builds a topology of interfaces, interface implementations and interface calls. SOA is a relationship of services and service consumers, both software modules large enough to represent a complete business function. So, SOA is about reuse, encapsulation, interfaces, and ultimately, agility."[142]

W3C:

„[A SOA is] a set of components which can be invoked, and whose interface descriptions can be published and discovered."[143]

Bei der Betrachtung dieser beiden Definitionen wird deutlich, wie unterschiedlich das Verständnis des Architekturkonzeptes der SOA ausgeprägt ist. Darüber hinaus erscheint es schwierig, eine klare Abgrenzung des Konzeptes gegenüber anderen architektonischen Ansätzen vorzunehmen, wie z. B. gegenüber der komponentenorientierten Softwareentwicklung.

Dieser Mangel eines einheitlichen Verständnisses um den Gegenstand und die Besonderheiten des Architekurkonzepts der SOA ist nicht nur für die wissenschaftliche Auseinandersetzung mit dem Architekturkonzept, sondern auch für seine praktische Umset-

---

[139] Zur komponentenorientierten Softwareentwicklung vgl. z. B. Crnkovic /Components/, Adler /Component Software/.

[140] Vgl. beispielsweise Hauser, Löwer /Web Services/ S. 13-15 & S. 25; Schlamann /Service Orientation/ S. 5-13; Dostal, Jeckle /Service-orientierte Architektur/ S. 53.

[141] Vgl. Kapitel 1.

[142] McCoy, Natis /Mainstream/ S. 1.

[143] W3C /Glossary/ S. 12; Gold u. a. /Understanding/ S. 72.

zung von großer Relevanz. Ein unklares oder gar unterschiedliches Verständnis einzelner Projektbeteiligter stellt beispielsweise ein hohes Risiko für die erfolgreiche Umsetzung einer SOA dar.[144] Insbesondere für das Architekturkonzept SOA ist dieses Risiko stark ausgeprägt, weil ein intensiver Dialog zwischen den Fachabteilungen und den Softwareingenieuren zur Erstellung eines Domänenmodells[145] erforderlich ist. Und die Entwickler einer Unternehmens-IT – also i. d. R. alle Entwickler eines Unternehmens – betroffen sind. Eine zielführende Kommunikation erscheint jedoch ohne ein gemeinsames Verständnis oder zumindest ohne ein Bewusstsein über unterschiedliche Ansichten nicht möglich.

2.2.2 Arbeitsdefinition SOA

In dieser Arbeit soll eine Definition verwendet werden, welche sowohl für Wissenschaftler als auch für Praktiker die Grundlage für ein gemeinsames Verständnis schafft.[146] Dabei soll insbesondere herausgearbeitet werden, worin die Besonderheiten des Architekturkonzeptes bestehen. Als Ausgangspunkt für die weitere Erläuterung wird die folgende Definition des Begriffes der SOA verwendet:

Eine SOA ist ein Konzept für eine Softwarearchitektur, in dem Funktionen in Form von wiederverwendbaren, technisch voneinander unabhängigen und fachlich lose gekoppelten Services implementiert werden. Services können unabhängig von zugrunde liegenden Implementierungen über wohldefinierte und veröffentlichte Serviceschnittstellen aufgerufen werden. Serviceinteraktion findet über eine dafür vorgesehene Kommunikationsinfrastruktur statt. Mit einer SOA werden insbesondere die Gestaltungsziele der Geschäftsprozessorientierung, der Wandlungsfähigkeit, der Wiederverwendbarkeit und der Unterstützung verteilter Softwaresysteme verbunden.

---

[144] Mertens stellt dies für unklare Verständnisse und ungenaue begriffliche Abgrenzungen für die Wirtschaftsinformatik im Allgemeinen dar. Vgl. Mertens /Gefahren/ S. 1744-1745.

[145] Vgl. Kapitel 2.2.5.2.

[146] Vgl. zur Herleitung der Arbeitsdefinition vgl. Kapitel 2.2 sowie Oey, Wagner, Rehbach, Bachmann /Einführende Darstellung/ S. 197-220; Die Praxistauglichkeit dieser Definition zeigt sich beispielsweise darin, dass diese Definition vom Arbeitskreis ‚Serviceorientierte Architektur' der Gesellschaft für Informatik e. V. als Definition einer SOA verwendet wird. Auch Diskussionen mit Experten und im Internet auf Wikipedia.de haben dies verdeutlicht. (Dort ist die Definition seit Mai 2005 bis auf unbedeutende – teilweise durch den Autor dieser Arbeit vorgenommene – Änderungen stabil.)

Diese Definition sowie die folgenden Erläuterungen stellen die grundlegende Struktur und die wichtigsten Gestaltungsziele einer SOA dar. Dabei wird von konkreten Technologien abstrahiert.[147] Die Definition ist Grundlage für die weitere Ausarbeitung und Argumentation.

2.2.3 SOA als unternehmensweites Architekturkonzept

Der Ausführung von Kapitel 2.1 folgend, kann eine Softwarearchitektur nach unterschiedlichen Betrachtungsebenen unterschieden werden. So kann eine SOA zum einen dazu eingesetzt werden, im Anwendungskontext neue Funktionalität zu realisieren und in die betriebliche Datenverarbeitung einzubinden; zum anderen ermöglicht sie im Unternehmenskontext z. B. die Integration bestehender heterogener Anwendungssysteme. Sie unterstützt somit auch Integrationskonzepte, wie die Enterprise Application Integration[148] oder die Business-to-Business Application Integration.[149]

In dieser Arbeit wird SOA insbesondere aus der Perspektive des unternehmensweiten Architekturkonzepts[150] beschrieben und analysiert. Aus dieser Perspektive wird die Datenverarbeitung an den Geschäftsprozessen der Unternehmen ausgerichtet. Es ist gerade diese Möglichkeit der unmittelbaren Verknüpfung der betrieblichen Abläufe und der sie unterstützenden Informationssysteme, die oftmals als entscheidende Stärke des Konzeptes angesehen wird.

2.2.4 Struktur einer SOA

Der oben dargestellten Definition des Begriffs ‚Softwarearchitektur' von Bass, Clements und Kazman[151] folgend, determiniert sich die Struktur einer SOA aus Softwarekomponenten, den extern sichtbaren Eigenschaften dieser und den Beziehungen zwischen den Softwarekomponenten als auch der Beziehungen der Softwarekomponenten zum Umfeld. Eine SOA besitzt als Softwarekomponenten Services[152], die in Kapitel

---

[147] Vgl. Kapitel 1.3.3.

[148] Kapitel 2.3.1. Für eine umfassende Darstellung vgl. Linthicum /Next Generation/.

[149] Für eine Einführung vgl. Linthicum /B2B/.

[150] Vgl. Kapitel 2.2.3.

[151] Vgl. Kapitel 2.1.

[152] Zum Begriff ‚Service' vgl. Kapitel 1.4.

2.2.4.1 vorgestellt werden. Nach außen sichtbare Eigenschaften der Services werden von Serviceschnittstellen repräsentiert, die in Kapitel 2.2.4.2 erläutert werden. Die Beziehungen der Services werden in Kapitel 2.2.4.3 vorgestellt.

2.2.4.1 Services

Die Services sind die zentralen Bestandteile einer Software, die nach den Prinzipien einer SOA implementiert wird. Sie sind die Softwarekomponenten, in denen bestimmte, klar abgegrenzte und eigenständig nutzbare Funktionen (im Folgenden Dienstleistungen genannt) implementiert werden und die von anderen Services genutzt werden können.[153]

Im Folgenden wird dargelegt, wie der Funktionsumfang von Services ausgeprägt sein kann, dass Services auf dem Prinzip des Versteckens ihrer Komplexität beruhen und wie sich Services von Komponenten eines herkömmlichen Software-Komponentenmodells unterscheiden.

Funktionsumfang von Services

Der Funktionsumfang[154] einzelner Services kann unterschiedlich ausgeprägt sein. Es wird zwischen niedrigem und hohem Funktionsumfang unterschieden.[155] Ein Service mit niedrigem Funktionsumfang bietet einfache Dienstleistungen an, während Services mit hohem Funktionsumfang komplexe Dienstleistungen zur Verfügung stellen. Einen allgemeingültigen Funktionsumfang für Services festzulegen erscheint nicht sinnvoll, weil dieser für jeden Service fallabhängig zu wählen ist.[156] Grundlagen zur Entscheidung, wie niedrig oder hoch der Funktionsumfang eines Services gewählt werden soll,

---

[153] Vgl. Banke, Krafzig, Slama /Enterprise SOA/ S. 59-60; Brandner u. a. /Architecture/ S. 144; Natis /SOA/ S. 23; Sholler /Resuable Enterprise Services/ S. 2, Carlson, Tyomkin /Good Design/ S. 14; Natis, Schulte /Introduction/ S. 1-3.

[154] In diesem Kontext wird auch von der ‚Granularität' von Services gesprochen. Vgl. Gallas /Enterprise Service Integration/ S. 187; Krüger, Seelmann-Eggebert /IT-Architektur/ S. 101. Aier, Schönherr /Flexibilisierung/ S. 5. Der Duden definiert ‚Granularität': Gra|nu|la|ri|tät, die; -, -en (EDV Anzahl von Untergliederungen eines Elements) Vgl. Wermke, Kunkel-Razum, Scholze-Stubenrecht /Duden/ S. 434.

[155] Vgl. META Group /Approaches/ S. 3.

[156] Einige Autoren geben den Ratschlag, eher hohen Funktionsumfang für einzelne Services zu wählen. Begründet wird dies damit, dass durch diese Maßnahme bei nicht eindeutig festlegbarem Funktionsumfang mit höherer Wahrscheinlichkeit eine hohe Abdeckung eines Geschäftsprozesses erreicht wird. Vgl. z. B. Brandner u. a. /Architecture/ S. 144; Sholler /Resuable Enterprise Services/ S. 2. Zur Diskussion des Funktionsumfangs vgl. Kapitel 3.2.2.2.1, 3.2.2.2.2, 3.2.2.6.1 und 4.3.1.

sind die zu unterstützenden Geschäftsprozesse, die daraus ableitbaren Dienstleistungen und die Wichtigkeit einzelner Gestaltungsziele.[157]

Einfluss auf den Funktionsumfang eines Services hat auch das Ausmaß der Schachtelungstiefe von Services. Zur Erreichung der später vorgestellten Gestaltungsziele kann die Einführung einer mehrfachen Verschachtelung von Services unter Umständen sinnvoll sein. So könnte eine Ebene von Infrastrukturservices[158] eingeführt werden, mit dem Ziel, die Wiederverwendung gebräuchlicher Dienstleistungen zu erhöhen.[159] Solche Infrastrukturservices bieten nur geringen Funktionsumfang, welcher allerdings in der gesamten Softwareumgebung sehr häufig von unterschiedlichen Geschäftsprozessen verwendet wird. Beispielsweise sind hier Ein-/Ausgabefunktionen (wie z. B. die Formatierung eines einheitlichen Geschäftsbriefes) oder Datenbankoperationen (wie z. B. Änderungen von Kundendatensätzen) häufig eingesetzte Funktionen für Infrastrukturservices. Solche Infrastrukturservices können von einem anderen Service in einem so genannten ‚zusammengesetzten Service' verwendet werden.[160]

Verstecken der eigenen Komplexität

Services beruhen auf dem Konzept des Versteckens der eigenen Komplexität, indem sie ihrem Umfeld nicht preisgeben, wie ihre Funktion implementiert ist (Black-Box-Prinzip).[161] Die Implementierung einzelner Services ist somit unabhängig von zugrunde liegenden Programmierparadigmen und Programmiersprachen und kann innerhalb einer SOA, also auch innerhalb einer Unternehmens-IT, durchaus unterschiedlich sein. Services können sogar auf Schnittstellen bestehender Anwendungssysteme aufsetzen und somit deren Integration ermöglichen, ohne dass dies den sonstigen Dienstleistungsnehmern bekannt ist. Diese Vorgehensweise folgt dem Prinzip, dass es den Servicenehmern in einer SOA prinzipiell unwichtig sein soll, wie eine Dienstleistung erbracht wird, solange dies im benötigten Rahmen effizient und effektiv geschieht.

---

[157] Vgl. Kapitel 4.2.

[158] Vgl. Banke, Krafzig, Slama /Enterprise SOA/ S. 65; Yang /SOA/ S. 9.

[159] In diesem Fall würde man davon ausgehen, dass Services, die Funktionen der Infrastruktur anbieten, durch viele andere Services genutzt werden (müssen).

[160] Vgl. Natis, Schulte /Introduction/ S. 1; Sholler /Resuable Enterprise Services/ S. 1; Lublinsky, Tyomkin /Dissecting SOA/ S. 54-55.

[161] Vgl. Woods /Enterprise Services Architecture/ S. 25; Natis, Schulte /Introduction/ S. 1; Tannhäuser, Umek /Architekturmanagement/ S. 66.

2.2.4.2 Serviceschnittstellen

Eine Schnittstelle ist eine Softwarekomponente, welche zwei oder mehrere Services miteinander zu dem Zweck verbindet, Informationen von der einen zur anderen zu übermitteln.[162] In einer SOA wird eine Schnittstelle eindeutig einem Service zugeordnet und stellt Dienstleistungsnehmern die Leistung bereit, Informationen mit diesem Service austauschen zu können. Um dies zu unterstreichen wird in dieser Arbeit der Begriff Serviceschnittstelle verwendet.

In einer Serviceschnittstelle sind die nach außen sichtbaren Eigenschaften eines Services gebündelt. Die Serviceschnittstellen sind somit die einzigen Softwarekomponenten eines Services, über die Dienstleistungen abgerufen werden können.[163] Daher müssen die Serviceschnittstellen eines Services genügend Informationen und Kommunikationswege bereitstellen, so dass der entsprechende Service verwendet und ggf. identifiziert werden kann, ohne dass ein Dienstleistungsnehmer die Implementierung betrachten müsste, um Informationen (z. B. bzgl. der Dienstleistung) zu erhalten.[164]

Verhältnis Service zu Serviceschnittstelle

Ein Service kann durchaus mehrere Serviceschnittstellen besitzen, jedoch bezieht sich eine Serviceschnittstelle immer auf nur einen einzigen Service.[165] Eine Serviceschnittstelle bietet die Dienstleistung eines Services für genau eine Aufrufart an. Um verschiedene Aufrufarten für die Dienstleistung eines Services anzubieten, müssen mehrere Serviceschnittstellen verwendet werden. Z. B. könnte ein (in diesem Fall Infrastruktur-)Service, der Aktienkurse zu Wertpapierkennzahlen zurückgibt, über eine XML basierte Schnittstelle und über eine Schnittstelle aufgerufen werden, die CSV-Dateien entgegen-

---

[162] Vgl. IEEE /Glossary/ S. 41; In der Definition des IEEE wird ‚Schnittstelle' je nach Perspektive in vier Definitionen geteilt: 1. Eine gemeinsame Grenze, über welche Informationen ausgetauscht werden. 2. Eine Hardware- oder Softwarekomponente, die zwei oder mehrere weitere Komponenten miteinander verbindet mit dem Zweck, Informationen von der einen zur anderen zu übermitteln. 3. Die Verbindung von zwei oder mehr Komponenten, für den Zweck der Informationsweitergabe von einer zur anderen. 4. Dient als Verbindung oder als Verbindungskomponente wie in 2. Alle Perspektiven teilen das gemeinsame Verständnis, dass Informationen weitergegeben werden. Übertragen auf das Konzept serviceorientierter Architekturen entsprechen Schnittstellen der vorgestellten Darstellung.

[163] Vgl. Chappell /ESB/ S. 2; Natis, Schulte /Introduction/ S. 1.

[164] Vgl. Natis, Schulte /Introduction/ S. 2 und Natis /SOA/ S. 23.

[165] Vgl. Sholler /Resuable Enterprise Services/ S. 1.

nimmt. Die Serviceschnittstellen liefern Rückantworten ebenso in einem definierten Format (vgl. Abb. 2-2).

Versionierung von Schnittstellen

Die Möglichkeit der Versionierung von Schnittstellen stellt eine Möglichkeit dar, einen Nachteil der komponentenbasierten Entwicklung zu kompensieren. Der Nachteil ist, dass Komponenten i. d. R. nicht an neue Anforderungen angepasst werden können, weil dies direkte Auswirkungen auf die bisherigen Nutzer einer Komponente hat.[166] Mit der Möglichkeit, unterschiedliche Versionen von Schnittstellen zu erstellen, können Nutzer eines Services weiterhin die bisher genutzte Serviceschnittstelle mit den entsprechenden bisher genutzten Eigenschaften verwenden.

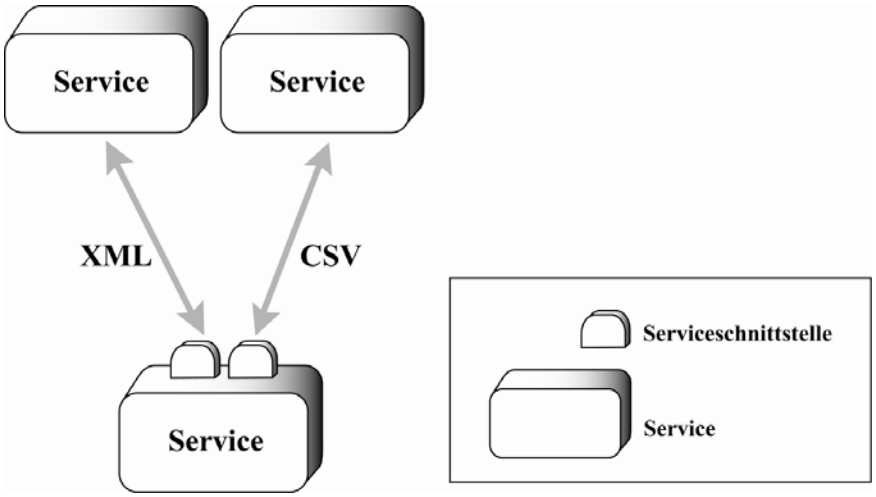

Abb. 2-2: Service & Serviceschnittstelle

2.2.4.3 Beziehungen der Services

Die Beziehung von Services untereinander entspricht der einer Geber-Nehmer-Beziehung. Services nehmen bei der Kommunikation die Rolle eines Servicegebers und/oder die eines Servicenehmers an.[167] Das Annehmen einer Rolle durch einen Servi-

[166] Vgl. Sommerville /Software Engineering/ S. 323.

[167] Zur Unterstreichung, dass ein Service beide Rollen annehmen kann, oder um eine Beziehung zwischen zwei Services klar darstellen zu können, kann ein solches Rollenverständnis angewendet wer-

ce muss nicht alternierend erfolgen und nicht alle Services müssen beide Rollen annehmen.

Die Beziehungen der Services wird im Folgenden beschrieben durch (1) den Kopplungsgrad, (2) die Art der Verbindung, (3) die Ausprägung der Beziehung und (4) durch den Einsatz einer Kommunikationsinfrastruktur, welche ein Service-Repository, ein Service-Registry und einen Service-Bus umfassen kann.

Kopplungsgrad

Die Beziehung von Services lässt sich über den Kopplungsgrad der Services zueinander beschreiben: Services sind zueinander lose gekoppelt.[168] Kopplung wird vom IEEE als das Ausmaß definiert, in dem Softwarekomponenten voneinander abhängig sind.[169] Die Abhängigkeit einer Kopplung zwischen Services kann sich auf folgende **drei Ebenen** beziehen:[170]

1. wie Services gefunden werden,
2. wie permanent die Informationen des Zustands eines Services (z. B. die Adresse) bei einem Dienstleistungsnehmer verfügbar ist und
3. wie die Kommunikation bzw. der Aufruf stattfindet.

Lose gekoppelt bedeutet, dass der Kopplungsgrad niedrig ist. Für die drei Ebenen bedeutet dies, dass im ersten Fall ein Mediator eingesetzt wird, um Dienstleistungen und die entsprechenden Services zu finden, dass im zweiten Fall das Feststellen des Ausführungsortes des Services vor dem Aufruf dynamisch erfolgt und für den dritten Fall, dass die Kommunikation über einen Mediator läuft.

Die lose Kopplung der Services kann darüber hinaus aus **zwei Perspektiven** betrachtet werden:[171]

---

den. Die Unterscheidung zwischen den Rollen Servicegeber und Servicenehmer ist jedoch nicht zwingend, da es sich durchaus um eine wandelnde Rollenzuordnung eines Services handeln kann.

[168] Vgl. z. B. Dostal, Jeckle /Service-orientierte Architektur/ S. 53; Natis, Schulte /Introduction/ S. 2.

[169] Vgl. IEEE /Glossary/ S. 22.

[170] Vgl. Banke, Krafzig, Slama /Enterprise SOA/ S. 62.

[171] Vgl. zu der Unterscheidung der losen Kopplung Banke, Krafzig, Slama /Enterprise SOA/ S. 62-64; Natis, Schule /Introduction/ S. 2-4; Wang u. a. /Integrated Quality of Service/ S. 3.

1. Aus einer *fachlichen Perspektive* werden Services so gestaltet, dass diese abgegrenzte und eigenständig nutzbare Funktionen erfüllen.
2. Aus einer *technischen Perspektive* werden die Services so gestaltet, dass ihre Implementierung unabhängig voneinander ist. Das bedeutet, dass eine technische Änderung eines Services vorgenommen werden kann, ohne dass die Anpassung der Implementierung anderer Services notwendig wird.

Der Kopplungsgrad von Services zu anderen Softwaresystemen hingegen kann in bestimmten Fällen hoch sein. Dies ist dann der Fall, wenn Dienstleistungen genutzt werden müssen, die nicht über Services implementiert wurden bzw. werden und somit keine Verbindung zu einem Service genutzt werden kann. Z. B. sind Services, die Daten aus Datenbanken abrufen oder Funktionen von Standardsoftware wie z. B. SAP/R3 nutzen, i. d. R. zu der entsprechenden Datenbank bzw. zu der entsprechenden Standardsoftware stark gekoppelt.

Das Konzept der losen Kopplung unterscheidet eine SOA zu Konzepten wie dem Client-Server-Konzept oder monolithischen Application-Servern.[172] In solchen Systemen wird i. d. R. ein sehr hoher Kopplungsgrad erreicht.[173]

Art der Verbindung

Die Beziehung von Services zueinander kann verbindungsorientiert oder verbindungslos gestaltet sein. Die Beziehung zwischen Services ist verbindungsorientiert, wenn zu dem Servicegeber eine direkte Verbindung besteht;[174] verbindungslos ist eine Beziehung dann, wenn ein Service eine Nachricht an einen anderen Service sendet und nicht weiß, ob die Nachricht empfangen oder verarbeitet wurde und ggf. sogar nicht den Empfänger kennt.

Ausprägung der Beziehung

Die Ausprägung einer Beziehung von Services zur Laufzeit kann synchron oder asynchron gestaltet sein. Synchrone Beziehungen erwarten auf eine Serviceanfrage eine di-

---

[172] Vgl. zum folgenden Absatz Chappell /ESB/ S. 2; Natis, Schulte /Introduction/ S. 1-2.

[173] Vgl. Zahavi /Integration/ S. 12-14.

[174] Vgl. zum folgenden Absatz Keller /EAI/ S. 78-80.

rekte Rückantwort, bevor der Servicenehmer in seiner Tätigkeit weiterarbeiten kann;[175] eine asynchrone Beziehung liegt dann vor, wenn ein Servicenehmer nicht auf eine Antwort des Servicegebers wartet, sondern weiterarbeitet. Einige Autoren vertreten die Meinung, dass eine SOA nur synchrone Beziehungen aufweisen sollte.[176] Eine solche Einschränkung wird in dieser Arbeit allerdings nicht getroffen, weil im Unternehmen Geschäftsprozesse asynchron ablaufen können,[177] und eine SOA sollte in der Lage sein, solche asynchronen Abläufe abzubilden.

Kommunikationsinfrastruktur

Die Kommunikation zwischen den Services wird über eine Kommunikationsinfrastruktur durchgeführt.[178] Diese dient dazu, die Services in die Lage zu versetzten, andere Services zu finden und die Übermittlung von Nachrichten zwischen Services zu unterstützen. Service-Repository, Service-Registry und Service-Bus werden unter dem Begriff Kommunikationsinfrastruktur zusammengefasst, weil diese zusammen diese Anforderungen erfüllen.

Service-Repository

Damit andere Services feststellen können, wie ein Service aufzurufen ist und welche Daten von einem Service erwartet werden, müssen die Schnittstelleninformationen eines Services veröffentlicht sein.[179] Die Veröffentlichung von Informationen zu den Serviceschnittstellen ebenso wie zu Services und den durch diese bereitgestellten Dienstleistungen, geschieht in einem gemeinsamen Service-Repository[180].[181] Software-Entwickler nutzen ein Service-Repository während der Implementierung eines Services, um – soweit notwendig – ein eindeutiges Kennzeichen zu erlangen und die Informationen des

---

[175] Vgl. zum folgenden Absatz Keller /EAI/ S. 76-80; Schlamann /Service Orientation/ S. 6.

[176] Vgl. Natis, Schulte /Introduction/ S. 2, Natis /SOA/ S. 25.

[177] Es kann sogar unterstellt werden, dass in Unternehmen mit arbeitsteiliger Aufgabenteilung sehr viele Geschäftsprozesse asynchron ablaufen, weil andernfalls eine Blockierung (vgl. Keller /EAI/ S. 78) einer Stelle auftreten würde.

[178] Vgl. Barry /Service-Oriented Enterprise Architecture/ S. 7.

[179] Vgl. Natis, Schulte /Introduction/ S. 2; Natis /SOA/ S. 23.

[180] Der Begriff Service Repository leitet sich aus dem Begriff Software Repository ab. Ein Software Repository ist eine Bibliothek, in welcher Information zu Software archiviert wird, vgl. IEEE /Glossary/ S. 71.

[181] Vgl. Natis, Schulte /Introduction/ S. 2; Banke, Krafzig, Slama /Enterprise SOA/ S. 60-64.

erstellten Services zu veröffentlichen.[182] Servicenehmer können die angebotene Dienstleistung über dieses Service-Repository auffinden und den entsprechenden dienstleistungsanbietenden Service aufrufen. Ohne ein Service-Repository können Informationen über Dienstleistungen nicht veröffentlicht und folglich auch nicht genutzt werden.[183]

Service-Registry

Das Auffinden eines Services wird darüber hinaus durch eine Service-Registry unterstützt. Diese setzt auf dem Service-Repository auf und ermöglicht die Lokalisierung eines Services (vgl. Abbildung 2).[184] Ein Service-Registry bietet in diesem Fall weitere Funktionen an, mittels denen Dienstleistungen aufgefunden werden können. So wird in der Literatur diskutiert, wie eine semantische Beschreibung einer nachgefragten Dienstleistung mittels eines Service-Registry so verarbeitet werden kann, dass die tatsächlich gewünschte Dienstleistung automatisch vermittelt werden kann.[185]

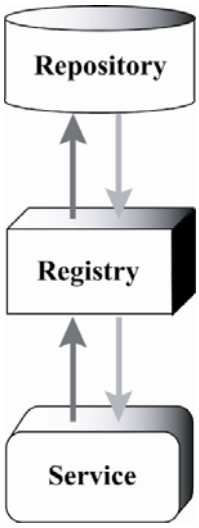

Abb. 2-1: Registry & Repository

[182] Vgl. Banke, Krafzig, Slama /Enterprise SOA/ S. 63.

[183] Vgl. Sholler /Governance/ S. 2.

[184] Dabei wird angestrebt, das Service-Registry in die Lage zu versetzen, Anfragen zu Dienstleistungen semantisch auszuwerten und selbstständig einen passenden Service auszuwählen. Vgl. Dostal, Jeckle /Service-orientierte Architektur/ S. 55; W3C /Glossary/ S. 68-74.

[185] Vgl. Banke, Krafzig, Slama /Enterprise SOA/ 60-61; Dostal, Jeckle, Kriechbaum /Semantik/ S. 46-47.

Service-Bus

Bei der Umsetzung von SOA wird darüber hinaus häufig ein Service-Bus eingesetzt. Dieser stellt Funktionen zur Verfügung, welche die Kommunikation zwischen Services unterstützen. Ein Service-Bus kann über das Übermitteln von Informationen hinaus weitere Aufgaben übernehmen, z. B. Verschlüsselung, Sicherheitsprüfung, Authentifizierung oder Lastverteilung.[186]

2.2.5 Gestaltungsziele einer SOA

Mit der Erläuterung der Struktur einer SOA wurden die grundlegenden informationstechnischen Bestandteile des Architekturkonzeptes eingeführt. Darauf aufbauend sollen im Weiteren die wesentlichen Ziele erläutert werden, die mit einer SOA verfolgt werden. Ihr Verständnis ist notwendig, um aus der abstrakten Beschreibung der Struktur konkrete Softwarearchitekturen abzuleiten, die der Idee der Serviceorientierung gerecht werden.

In der Praxis kann es durchaus Fälle geben, in denen die stringente Verfolgung aller im Folgenden genannten Gestaltungsziele nicht notwendig erscheint bzw. durch das Management nicht angestrebt wird. Dies kann z. B. durch eine geringe Ausbreitung des Konzepts in der Unternehmens-IT ausgelöst werden, wie es in Pilotprojekten oder Machbarkeitsstudien zu vermuten ist.

2.2.5.1 Betriebliche Anforderungen an die Gestaltung einer SOA

Die Gestaltung einer SOA im Sinne eines unternehmensweiten Architekturkonzeptes basiert nicht nur auf technischen Zielen, sondern bezieht sich auf die betrieblichen Anforderungen an die Datenverarbeitung.

Infolge der Globalisierung der Märkte und der zunehmenden weltweiten informationstechnischen Verknüpfung stehen die Unternehmen neuen strategischen Herausforderungen gegenüber.[187] Meist wird neben einer hohen Leistungsfähigkeit auch eine hohe An-

---

[186] Die Anbieter von Integrationsplattformen beispielsweise erweitern ihre Plattformen um Funktionen, die speziell an die Anforderungen einer SOA angepasst wurden. Solche Funktionen werden typischerweise in einem ‚Service-Bus' angeboten und dienen hauptsächlich als Kommunikationskanal. Vgl. Chappell /ESB/ S. 42.

[187] Vgl. Kapitel 3.1.4.2.3.

passungsfähigkeit an sich ändernde Bedingungen als wesentlich betrachtet, um erfolgreich im Wettbewerb bestehen zu können. Diesen strategischen Vorgaben gilt es im Rahmen der organisatorischen Gestaltung durch geeignete Abläufe und Strukturen zu entsprechen.[188] Einer Orientierung an Geschäftsprozessen wird deswegen seit einiger Zeit besondere Bedeutung beigemessen.[189]

Ein Geschäftsprozess kann als eine Folge von logisch zusammengehörigen Aktivitäten definiert werden, die einen Beitrag zur Wertschöpfung des Unternehmens leisten.[190] Häufig basieren diese Abläufe auf einer Interaktion mit den Kunden. Im Zuge der geschäftsprozessorientierten Gestaltung der Organisation gilt es, einzelne Arbeitsschritte so abzugrenzen und zu zusammengehörenden Prozessen zu kombinieren, dass diese einen möglichst großen Beitrag zur Wertschöpfung des Unternehmens und ggf. zum Nutzen des Kunden leisten.[191] Wichtig ist ferner die prozessinterne sowie prozessübergreifende Harmonisierung der Abläufe in zeitlicher und räumlicher Hinsicht zur Gewährleistung einer hohen Effektivität und Effizienz der Leistungserbringung.[192] Da die Geschäftsprozesse häufig nicht nur ein Unternehmen, sondern die Interaktion verschiedener Unternehmen betreffen, ist es dabei von Bedeutung, nicht nur eine unternehmensinterne, sondern auch eine unternehmensübergreifende Perspektive einzunehmen.

Durch die Gestaltung der Informationssysteme wird angestrebt, die im Rahmen der definierten organisatorischen Abläufe und Strukturen erforderliche Datenverarbeitung bestmöglich zu unterstützen und dabei deren Anpassbarkeit an sich ändernde strategische und organisatorische Gegebenheiten soweit wie möglich zu gewährleisten.[193] Der hohe Informationsbedarf eines Unternehmens, die wachsenden Anforderungen an die Geschwindigkeit und Qualität der Datenverarbeitung sowie die Notwendigkeit der Anpassbarkeit stellen eine große Herausforderung für die Gestaltung der informationstechnischen Strukturen dar – eine Herausforderung, zu deren Bewältigung das Konzept der

---

[188] Vgl. Österle /Integration/ S. 3; Kirchmer /Einführung/ S. 3.

[189] Vgl. Kirchmer /Einführung/ S. 8-11. Die Geschäftsprozessorientierung auf der Ebene der organisatorischen Gestaltung kommt beispielsweise in der praktischen Bedeutung der Konzepte des Business Process Reengineering (BPR) oder des Supply Chain Management (SCM) zum Ausdruck. Vgl. hierzu beispielsweise Davenport, Short /Business Process Redesign/.

[190] Vgl. Stahlknecht, Hasenkamp /Wirtschaftsinformatik/ S. 228.

[191] Vgl. Kirchmer /Einführung/ S. 1-10.

[192] Vgl. Frese /Organisation/ S. 7.

SOA insbesondere bei der Verfolgung von vier Gestaltungszielen geeignet erscheint: der Orientierung an Geschäftsprozessen, der Unterstützung der Wandlungsfähigkeit, der Unterstützung der Wiederverwendbarkeit und der Unterstützung verteilter Systeme.

2.2.5.2 Orientierung an Geschäftsprozessen

Die generelle Anforderung an die betrieblichen Informationssysteme lautet, wie beschrieben, die im Rahmen der definierten organisatorischen Abläufe und Strukturen erforderliche Datenverarbeitung bestmöglich zu unterstützen.[194] Als Konsequenz wird somit der Orientierung an Geschäftsprozessen eine besondere Bedeutung eingeräumt.[195] Beachtet werden muss hierbei auch, dass im Gesamtkontext eines Unternehmens Beziehungen nicht nur innerhalb eines Geschäftsprozesses, sondern auch zwischen Geschäftsprozessen bestehen.[196]

Deswegen muss versucht werden, die Abgrenzung von Services so vorzunehmen, dass sinnvolle Teilschritte von Geschäftsprozessen abgebildet werden, um dadurch unternehmerische Tätigkeiten durch die Unternehmens-IT unterstützen zu können. SOA soll die Möglichkeit schaffen, einzelne Services im Sinne von ‚Prozessbausteinen' mit geringem Aufwand zu Anwendungen zusammenzusetzen (dieser Prozess wird Orchestrierung[197] genannt). Auch soll sie in der Lage sein, automatisierte Interaktionsfolgen der Services zu realisieren (sog. Choreographien).[198] Services können dabei zu bestehenden Choreographien hinzugefügt, entfernt oder neu angeordnet werden. Das dabei verwendete Modell der Geschäftsprozesse eines Unternehmens, an welches die Service-Choreographien angepasst werden, wird Domänenmodell[199] genannt.

Durch die Orientierung an den Geschäftsprozessen bei der Abgrenzung und Orchestrierung der Services wird versucht, die Abläufe der unternehmerischen Leistungserbrin-

---

[193] Vgl. Oxman, Smith /Structual Change/ S. 77; Hansen, Neumann /Wirtschaftsinformatik/ S. 523-524; Aier, Schönherr /Flexibilisierung/ S. 22.

[194] Vgl. Kapitel 2.2.5.1.

[195] Vgl. Kirchmer, Scheer /Change Management/ S. 2-3.

[196] Vgl. Kirchmer /Einführung/ S. 41.

[197] Vgl. zu diesem Absatz Nadhan /Service-Oriented Evolution/ S. 43-44.

[198] Vgl. Adam, McKendrick /Everything in Between/ S. 22.

[199] Vgl. Hofmeister, Nord, Soni /Software Architecture/ S. 4-5.

gung möglichst vollständig und durchgängig abzubilden.[200] Dieses Vorgehen soll auch zu einer höheren Klarheit der informationstechnischen Strukturen beitragen, weil sich diese nicht von den Strukturen der Geschäftsprozesse des Unternehmens unterscheiden.[201] Voraussetzung dafür ist ein genaues Verständnis der zu unterstützenden Geschäftsprozesse.[202] Die Modellierung von Services einer SOA orientiert sich folglich an fachlichen und nicht an technischen Anforderungen.[203]

2.2.5.3 Unterstützung der Wandlungsfähigkeit

Wandlungsfähigkeit bedeutet in diesem Kontext, dass geänderte und neue Anforderungen an die Datenverarbeitung mit möglichst geringem Aufwand umgesetzt werden können. Auf diese Weise wird dazu beigetragen, dass Geschäftsprozesse an veränderte strategische und operative Anforderungen angepasst werden können. In fachlicher Hinsicht wird die Wandlungsfähigkeit durch das ‚Denken in Services' und die modulare Sichtweise auf Geschäftsprozesse unterstützt; in technischer Hinsicht wird die Hoffnung geäußert, dass eine SOA der Forderung nach Wandlungsfähigkeit durch ihre beschriebene Struktur und die dabei angestrebte Modularisierung, Kapselung und lose Kopplung gerecht werden kann.

2.2.5.4 Unterstützung der Wiederverwendbarkeit

Eine hohe Wiederverwendbarkeit ist ein weiteres Ziel, das mit der Einführung einer SOA verfolgt wird. Durch eine hohe Wiederverwendbarkeit sollen der erforderliche Entwicklungs- und Wartungsaufwand verringert und eine konsistente Datenverarbeitung durch die Vermeidung der negativen Folgen von Redundanzen unterstützt werden.

Dabei können zwei Perspektiven unterschieden werden. Aus einer technischen Perspektive wird die Wiederverwendung von Teilen des Programmcodes angestrebt; aus einer fachlichen Perspektive unterstützt das ‚Denken in Services' die Wiederverwendung

---

[200] Vgl. Gallas /Life Cycle/ S. 235; Woods /Enterprise Services Architecture/ S. 25; Kirchmer /Einführung/ S. 28; Eine grundsätzliche Strukturierung von unternehmerischen Abläufen an Geschäftsprozessen wird von Lucas vorgestellt, vgl. Lucas /Information Technology/ S.128.

[201] Vgl. Tannhäuser, Umek /Architekturmanagement/ S. 70; Natis /SOA/ 24.

[202] Vgl. Sholler /Governance/ 1.

[203] Vgl. Tannhäuser, Umek /Architekturmanagement/ S. 70; Sholler /Resuable Enterprise Services/ S. 2; Banke, Krafzig, Slama /Enterprise SOA/ S. 67.

definierter Arbeitsschritte in verschiedenen Geschäftsprozessen. SOA strebt an, beide Perspektiven zu unterstützen.[204] Services und damit die durch sie angebotenen Funktionen sollen mehrfach für die Unterstützung verschiedener Geschäftsprozesse eingesetzt werden können.[205] Auch wird versucht, dadurch eine konsistente Umsetzung der entsprechenden Dienstleistung unternehmensweit zu gewährleisten. Wichtiges Ziel ist auch, eine geeignete Abgrenzung der durch einzelne Services zu erbringenden Dienstleistung zu erreichen[206] und Wiederverwendung auf der Ebene der (Teil-)Geschäftsprozesse zu ermöglichen, nicht nur auf der technischen Ebene der Softwarekomponenten.

2.2.5.5 Unterstützung verteilter Systemzugriffe

Große Unternehmen agieren vermehrt auf globaler Ebene:[207] Ihre Absatz- und Faktormärkte als auch ihre Niederlassungen (Zweigstellen, Beteiligungen, usw.) sind über die ganze Welt verteilt. Solche für Unternehmen heute übliche Bedingungen und organisatorische Strukturen stellen besondere Anforderungen an das Unternehmen. Mit dem Unternehmen ist aber gleichzeitig auch die Unternehmens-IT gefordert, mit den entsprechenden Anforderungen umgehen zu können.[208]

SOA verfolgt deswegen das Ziel, dass die Erbringung einer Dienstleistung vom Standort des Nachfragers und vom Standort des Anbieters einer Dienstleistung unabhängig ist. Z. B. sollte ein Mitarbeiter eines deutschen Unternehmens in der Zweigstelle in den USA in seinem Anwendungssystem auf Dienstleistungen zugreifen können, die für das gesamte Unternehmen über einen Service angeboten werden, der auf Hardware in Deutschland abläuft. Dass ein solcher verteilter Systemzugriff[209] stattgefunden hat, ist für den Anwender einer Unternehmens-IT, die nach Prinzipien einer SOA erstellt wurde, nicht ersichtlich.

---

[204] Vgl. Banke, Krafzig, Slama /Enterprise SOA/ S. 281; Sholler /Resuable Enterprise Services/ S. 2; Sholler /Case/ S. 1.

[205] Vgl. Lublinsky, Tyomkin /Dissecting SOA/ S. 54.

[206] Vgl. Huizen /SOA/ S. 25. Beispielsweise führt eine Struktur vieler Services mit niedrigem Funktionsumfang zu einer potentiell höheren Wiederverwendung einzelner Services als eine Struktur weniger Services mit hohem Funktionsumfang.

[207] Vgl. Kieser, Walgenbach /Organisation/ S. 284-285.

[208] Vgl. Sommerville /Sofware Engineering/ S. 622 sowie Kapitel 3.2.2.7.2 und 3.2.3.3.

[209] Vgl. Sommerville /Sofware Engineering/ S. 622.

Mit der Unterstützung verteilter Systemzugriffe wird das Ziel verfolgt, dass Unternehmen Anwendungen ihrer Unternehmens-IT redundant aufbauen, warten und einsetzen müssen, um den Anforderungen globaler Unternehmensstrukturen und globalen Handelns gerecht werden zu können.

2.3 Abgrenzung serviceorientierter Architekturen

Im Zusammenhang mit SOA werden häufig weitere Konzepte genannt. Um ein klares Verständnis zu schaffen, wo der Unterschied bzw. wo Gemeinsamkeiten zu finden sind, wird im Folgenden zu einigen dieser Konzepte eine Abgrenzung gezogen bzw. eine Erläuterung der Gemeinsamkeiten skizziert.

Die hier zur Abgrenzung betrachteten Konzepte sind die Enterprise Application Integration, die ereignisgesteuerte Architektur (bekannter unter dem Akronym EDA, event-driven architecture) und der Komponentenorientierung.

2.3.1 Abgrenzung zur Enterprise Application Integration

Enterprise Application Integration bedeutet nach Linthicum die uneingeschränkte gemeinsame Nutzung von Daten und Geschäftsprozessen zwischen allen miteinander verbundenen Anwendungen und Datenquellen in einem Unternehmen.[210] Allgemeiner kann Enterprise Application Integration so aufgefasst werden, dass eine umfassende Integration der im Unternehmen verwendeten IT-Systeme angestrebt wird.[211] Das angestrebte Ziel ist, Informationen und Anwendungen zu verbinden, um so eine reibungslose Unterstützung der unternehmerischen Tätigkeiten durch die Unternehmens-IT zu erreichen.

Bei der Durchführung einer Enterprise Application Integration (EAI) müssen somit unterschiedliche Softwareanwendungen über ihre jeweiligen technischen und strukturellen Infrastrukturen hinweg integriert werden.[212] Eine solche Integration stellt ein komplexes Problem dar, insbesondere da Lösungen nicht nur für Teile eines Unternehmens, z. B.

---

[210] Linthicum /Application Integration/ S. 3.

[211] Vgl. Carlson, Tyomkin /Good Design/ S. 17.

[212] Dettling /EAI/ S. 7.

für Abteilungen oder Profit-Center, sondern für das gesamte Unternehmen gefunden und umgesetzt werden müssen.[213]

Für die Umsetzung einer solchen Enterprise Application Integration stehen heute unterschiedliche technische und konzeptionelle Integrationslösungen bereit.[214] Da eine SOA eine Vorgehensweise ist, um Anwendungen auf einer Dienstleistungsebene zu verbinden, sieht Linthicum eine SOA als eine Form der Umsetzung einer Enterprise Application Integration an.[215] In der Literatur wird jedoch auch die Sichtweise vertreten, dass eine SOA eine ‚höhere Form' der Integration, aufgrund der Orientierung an Geschäftsprozessen, darstellt.[216] Beide Ansichten können vertreten werden, ohne dass sie sich widersprechen. In dieser Arbeit wird eine SOA als eine Alternative zur Durchführung einer EAI betrachtet.

2.3.2 Abgrenzung zur event-driven architecture (EDA)

Eine ereignisgesteuerte Architektur (event-driven architecture) wird definiert als eine Architektur, in der voneinander vollständig entkoppelte Softwarekomponenten[217] ununterbrochen einen Kommunikationskanal überwachen. Sie reagieren nur auf Nachrichten, die durch sie bearbeitet werden können.[218] Die wichtigsten Aspekte zur Unterscheidung zwischen SOA und EDA sind, dass Softwarekomponenten einer EDA einen beliebig großen Funktionsumfang aufweisen können,[219] nie direkt miteinander kommunizieren und jede Informationsweitergabe asynchron stattfindet.[220]

Da sowohl bei einer Softwarearchitektur nach serviceorientierten als auch nach ereignisgesteuerten Prinzipien eine Kommunikationsinfrastruktur Anwendung findet, ist

---

[213] Vgl. Linthicum /Next Generation/ S. 18: Linthicum behauptet, dass der Nutzen einer EAI mit der Größe des Integrationsproblems wächst.

[214] Vgl. Carlson, Tyomkin /Good Design/ S. 14.

[215] Vgl. Linthicum /Next Generation/ S. 18-20.

[216] Vgl. Carlson, Tyomkin /Good Design/ S. 14: „SOA takes EAI to the next level."

[217] Vgl. Natis, Schulte /Introduction/ S. 2.

[218] Natis /SOA/ S. 25.

[219] Dies kann sogar soweit führen, dass beispielsweise nach einer Fusion zweier Unternehmen die jeweilige Unternehmens-IT weiterverwendet und um jeweils eine Komponente erweitert wird, welche alle Aktivitäten als separate Nachricht über die gemeinsame Kommunikationsinfrastruktur sendet und gleichzeitig alle zur Weiterverarbeitung geeigneten Nachrichten sammelt.

[220] Vgl. zu diesem Absatz Natis, Schulte /Introduction/ S. 2; Natis /SOA/ S. 25.

grundsätzlich auch eine Kombination beider Architekturkonzepte möglich.[221] Im Kontext von SOA kann eine Kommunikation zwischen Services somit ereignisgesteuert umgesetzt werden. Dies ist z. B. dann sinnvoll, wenn ein Service eine asynchrone Kommunikation mit einem anderen Service nutzt. Eine solche Nachricht ist in einer SOA zwar zielgebunden, d. h., der Empfänger ist bekannt und wird durch den Service-Bus direkt zur Übermittlung der Nachricht angesprochen, die Nachricht kann aber auch als Ereignis aufgefasst werden, das die Bearbeitung einer Dienstleistung letztendlich auslöst.

Für die Analyse der Arbeit findet im Folgenden eine Fokussierung auf eine SOA nach der Definition des Kapitels 2.2.2 statt.

2.3.3 Abgrenzung zu Komponentenmodellen

Zu Softwarekomponenten bekannter Komponentenmodelle[222] wie DCOM, CORBA oder JavaBeans unterscheiden sich Services sowohl im Hinblick auf den Umfang und die Abgrenzung der durch sie erbrachten Funktion als auch im Hinblick auf ihre technische Gestaltung. So deckt der Funktionsumfang eines Services einen oder mehrere Arbeitsschritt(e) eines Geschäftsprozesses ab. Die Abgrenzung erfolgt somit unter fachlichen Gesichtspunkten und kann in einen unterschiedlichen Funktionsumfang resultieren.[223] Komponentenmodelle enthalten hingegen Komponenten, die nur mit Services vergleichbar sind, die einen niedrigen Funktionsumfang anbieten und i. d. R. keinen abgeschlossenen Geschäftsprozess abdecken, sondern auf rein funktionalen oder datenorientierten Anforderungen basieren.[224] Gegenwärtig wird vorgeschlagen, Komponenten eines Komponentenmodells nach fachlichen Anforderungen zu erstellen, z. B. mehrere Funktionen zu Komponenten zusammenzufassen oder komplette Programme, wie z. B. ein Tabellenkalkulationsprogramm, als Komponenten zur Verfügung zu stel-

---

[221] Vgl. Natis /SOA/ 25: „The real-time enterprise of 2008 will be both service-oriented and event-driven."

[222] Vgl. Gruhn, Thiel /Komponentenmodelle/ S. XII-XIII.

[223] Vgl. Aier, Schönherr /Flexibilisierung/ S. 5.

[224] Vgl. zum Aufbau komponentenorientierter Systeme Gruhn, Thiel /Komponentenmodelle/ S. 6, 13-14. Das Konzept der Enterprise Java Beans beispielsweise beinhaltet sog. *Entities*, die sich sehr eng an Datenstrukturen orientierten, weil sie (dem Konzept folgend) eine Abbildung von Datenbankstrukturen in Java-Klassen darstellen sollen. Vgl. Schmietendorf, Dimitrov, Dumke /JavaBeans/ S. 207-211

len.[225] Hinsichtlich der technischen Gestaltung basieren Komponentenmodelle auf einer konkreten Realisierungstechnologie (z. B. DCOM oder CORBA), während eine SOA unabhängig von konkreten technischen Lösungsmöglichkeiten, sowohl in Bezug auf Plattformen als auch auf die Implementierungstechnologien, umgesetzt werden kann.[226]

Ein weiteres Unterscheidungskriterium ist, dass bei der komponentenorientierten Entwicklung ausgehend von vorhandenen Komponenten entworfen und entwickelt wird.[227] Ein solcher Entwurf, bei dem ein Endprodukt aus einzelnen, vorhandenen Komponenten entsteht, kann weniger leistungsfähig sein als ein Entwurf, der auf ein konkretes Ziel ausgerichtet ist und die später enthaltenen Komponenten festlegen kann.[228] Dieser Nachteil wird bei der Entwicklung einer SOA dadurch vermieden, dass sich die Gestaltung an Geschäftsprozessen orientiert[229] und folglich auch konkrete unternehmerische Ziele verfolgt werden.

---

[225] Vgl. Sommerville /Software Engineering/ S. 321.

[226] Aus diesem Grund wird in diesem Beitrag bewusst von konkreten Technologien abstrahiert.

[227] Vgl. Sommerville /Software Engineering/ S. 322.

[228] Vgl. Sommerville /Software Engineering/ S. 322.

[229] Vgl. Kapitel 2.2.5.2.

3. Nutzenpotential einer SOA

Der erste Schritt zur Wirtschaftlichkeitsbewertung einer SOA ist die Bewertung des Nutzenpotentials.[230] Diesbezüglich wird in Kapitel 3.1 zunächst ein Qualitätsmodell zur Softwarearchitekturbewertung aufgestellt. In Kapitel 3.2 wird eine Bewertung des Nutzenpotentials für SOA anhand des aufgestellten Qualitätsmodells durchgeführt.

3.1 Qualitätsmodell für die Softwarearchitekturbewertung

Ein Qualitätsmodell und dessen Bestandteile (die Qualitätsattribute zur Bewertung eines Nutzenpotentials einer Softwarearchitektur) können unter verschiedenen Perspektiven erstellt werden.[231] In dieser Arbeit werden zur Softwarearchitekturbewertung zwei Perspektiven als relevant identifiziert: Die Perspektive der Software-Entwickler einer Unternehmens-IT und die Perspektive des Managements des Unternehmens. Die erste ist relevant, weil die Software-Entwickler mit der Erstellung und Wartung der Software beauftragt sind und ihre jeweiligen Aufgaben durch eine Softwarearchitektur beeinflusst werden; die zweite ist relevant, weil die Kernaufgabe des Managements betroffen ist: die Unternehmensführung. Unternehmensführung ist „die Gesamtheit derjenigen Handlungen (…), welche die Gestaltung und Abstimmung (Koordination) der Unternehmens-Umwelt-Interaktion im Rahmen des Wertschöpfungsprozesses zum Gegenstand haben, und diesen grundlegend beeinflussen.“[232] Die Unternehmens-IT soll das Unternehmen bestmöglich unterstützen[233] und hat dadurch Einfluss auf die Unternehmens-Umwelt-Interaktion und Wertschöpfungsprozesse. Die Gestaltung der Unternehmens-IT ist folglich eine Aufgabe des „oberen und obersten Managements“[234]. Weitere Perspektiven wie z. B. die der Kunden, Mitarbeiter und Lieferanten werden im Qualitätsmodell nicht explizit berücksichtigt, weil diese Gruppen zum großen Teil funktionale Anforde-

---

[230] Vgl. Kapitel 1.3.2.

[231] Vgl. Laprie /Dependability/ S. 6; McCall /Quality factors/ S. 959-960; Bass, Clements, Kazman /Software Architecture/ S. 281. Beispielsweise kann aus der Werkzeug-Perspektive das Qualitätsattribut Testbarkeit der Wartbarkeit untergeordnet werden, während es aus der Lebenszyklusperspektive dem Qualitätsattribut Produktevolvierbarkeit zugeordnet werden kann. Vgl. McCall /Quality factors/ S. 959-960.

[232] Macharzina /Unternehmensführung/ S. 38.

[233] Vgl. Kapitel 2.2.5.1.

[234] Macharzina erläutert, dass die Aufgabe der Unternehmensführung die Aufgabe des „oberen und obersten Managements“ ist. Macharzina /Unternehmensführung/ S. 38.

rungen stellen und dies im Folgenden nicht betrachtet wird.[235] Nicht-funktionale Anforderungen dieser Perspektiven werden im Qualitätsmodell indirekt berücksichtigt: Diese können als Qualitätsattribute der Software-Entwickler verstanden werden, weil die genannten Gruppen die ‚Kunden' der Software-Entwickler darstellen und ihre Anforderungen durch diesen Status in die Qualitätsattribute der Software-Entwickler einfließen.[236]

Durch die Fokussierung auf zwei Perspektiven wird das Qualitätsmodell zur Nutzenbewertung in zwei Qualitätsbereiche untergliedert: zum einen werden sog. technisch-fachliche und zum anderen sog. geschäftliche Qualitätsattribute aufgestellt. Die erstgenannten sind die Qualitätsattribute aus der Perspektive der Software-Entwickler einer Unternehmens-IT. Sie stellen grundlegend die bisher verwendeten Qualitätsattribute der (Wirtschafts-)Informatik zur Qualitätsbewertung von Software dar. Die letztgenannten repräsentieren die Perspektive des Managements eines Unternehmens und wurden in bisher verwendeten Qualitätsmodellen nicht betrachtet. Gegenwärtig ist eine Bewertung der Anforderungen an eine Unternehmens-IT aus der Unternehmensperspektive notwendig, weil die Unternehmens-IT über ihren Charakter als effizienzsteigerndes Instrument hinaus Bedeutung gewinnt und als Allgemeingut für Unternehmen angesehen werden kann.[237]

Im Kapitel 3.1.1 wird auf die Problematik und Notwendigkeit zur Erstellung eines Qualitätsmodells zur Softwarearchitekturbewertung eingegangen. Im Kapitel 3.1.2 wird der Rahmen des hier verwendeten Qualitätsmodells vorgestellt. Die (Unter-)Qualitätsattribute werden in den folgenden Kapiteln erarbeitet und entsprechende Arbeitsdefinitionen der Qualitätsattribute aufgestellt. Die Qualitätsattribute bilden zum einen eine technisch-fachliche Perspektive (Kapitel 3.1.3), zum anderen die geschäftliche Perspektive (Kapitel 3.1.4) auf die Qualität einer unternehmensweiten Softwarearchitektur ab.

---

[235] Vgl. zur Argumentation Kapitel 3.1.2.

[236] In dieser Arbeit verwendete Qualitätsattribute, welche in diese Kategorie fallen, sind beispielsweise Funktionserfüllung und Gebrauchsgerechtigkeit.

[237] Vgl. Carr /IT/ S. 44-47.

3.1.1 Problematik der Erstellung eines Qualitätsmodells zur Softwarearchitekturbewertung

In der Literatur konnte keine Sammlung allgemein anerkannter Qualitätsattribute zur Qualitätsbewertung von Software gefunden werden. Eine Erklärung dafür ist, dass Qualitätsattribute eines Qualitätsmodells die Ziele abdecken sollen, die von einem Unternehmen als wichtig angesehen werden.[238] Da Ziele eines Unternehmens zur Softwarebewertung unterschiedlich gewichtet werden können, lassen sich einzelne Qualitätsattribute und vielfältige unterschiedliche Zusammenstellungen dieser Qualitätsattribute zur Softwarebewertung vielfältig in der Literatur finden: Z. B. haben das Software Engineering Institute (SEI),[239] Mitarbeiter des SEI in unterschiedlichen Bewertungsprojekten[240] und die ISO als ISO-Standard 9126 Qualitätsmodelle (mit entsprechenden Qualitätsattributen) für Software veröffentlicht.

Ein Qualitätsmodell zur Bewertung einer Softwarearchitektur wird hier aufgestellt, weil in der Literatur keine Qualitätsmodelle mit dem Fokus der Bewertung von Softwarearchitekturen gefunden werden konnten. Es ist offensichtlich, dass solche Qualitätsmodelle, die als Spezialisierung der Qualitätsmodelle zur Bewertung von Software angesehen werden können, nicht einfach aufzustellen sind. Die Problematik zur Aufstellung eines Qualitätsmodells besteht in der Auswahl relevanter Qualitätsattribute.[241] Komplexe Gebilde weisen eine Vielzahl von Eigenschaften und Dimensionen auf, von denen nur eine begrenzte Anzahl für eine Bewertung relevant ist. Entscheidend ist daher, dass die für den jeweiligen Untersuchungszweck wichtigen Eigenschaften bzw. Dimensionen ausgewählt werden. Die Auswahl determiniert den Realitätsausschnitt, der in die anschließende Analyse eingeht. Eigenschaften, die bei der Festlegung der Dimensionen nicht erfasst worden sind, können bei nachfolgenden Analysten nicht mehr als wichtig erkannt und ihre Auswirkungen nicht diskutiert werden. Umgekehrt kann sich eine Argumentation auf Eigenschaften beziehen, die für eine Analyse nicht (mehr) relevant

---

[238] Vgl. zu diesem Absatz Kazman /Software Architecture/ S. 281 und Clements, Kazman, Klein /ATAM/ S. 9-11.

[239] Vgl. Barbacci u. a. /Quality Attributes/ S. 3-42.

[240] Vgl. Asundi, Kazman, Klein /Economic Considerations/ S. 20; Bass, Clements, Kazman /Software architecture/ S. 296-297; Clements, Kazman, Klein /ATAM/ S. 16-17; Asundi, Kazman, Klein /Economic Approach/ S. 20; Asundi, Kazman, Klein /Quantifying/ S. 299.

[241] Vgl. zum folgenden Absatz Kieser, Walgenbach /Organisation/ S. 71-73.

sind, jedoch im Qualitätsmodell enthalten sind, und dadurch ein Ergebnis verzerren. Die Auswahl der Qualitätsattribute ist deswegen ein kritisches Problem.

Das hier aufgestellte Qualitätsmodell beruht auf der Konsolidierung von vier Qualitätsmodellen zur Softwarebewertung und wird um weitere Qualitätsattribute ergänzt. Qualitätsattribute zur Bewertung von Software können auch zur Bewertung von *Softwarearchitekturen* angewendet werden, weil eine Softwarearchitektur den Gestaltungsrahmen von Software festlegt und dadurch direkten Einfluss auf die Qualität von Software hat.[242] Weitere Qualitätsattribute, die zur Architekturbewertung herangezogen werden können, konnten identifiziert werden. Zum einen haben Softwarearchitekturen einen anderen Fokus als Software, nämlich auf die Unterstützung der Softwareentwicklung als Werkzeug und Gestaltungsrichtlinie,[243] wohingegen Software der Unterstützung des Unternehmen dient;[244] zum anderen haben Softwarearchitekturen eine größere Tragweite im Vergleich zu Software, weil Softwarearchitekturen in unternehmensweiter Perspektive als Basis einer Infrastruktur für die gesamte Unternehmens-IT dienen,[245] wohingegen Software primär die Anwender und Auftraggeber zufrieden stellen muss.[246]

Die vier Qualitätsmodelle wurden nach den Kriterien ausgewählt, dass entweder eine Beachtung von Softwarearchitekturen im Qualitätsmodell (Raasch betrachtet Softwarearchitekturen bei der Aufstellung seiner Qualitätsattribute[247]) oder eine hohe Durchdringung und Akzeptanz der Qualitätsmodelle in Wissenschaft und Praxis vorliegt (diese sind ISO-Standard 9126[248], Boehm sowie McCall[249]). Diese Qualitätsmodelle entsprechen der bisherigen Sichtweise auf Qualitätsattribute vom Standpunkt der (Wirtschafts-)Informatik aus betrachtet. D. h., es wird sowohl eine technische als auch eine fachliche Sichtweise der Softwarequalität vertreten. Die Qualitätsattribute dieser Qualitätsmodelle

---

[242] Aus diesem Grund wurde auch Literatur gefunden, welche zur Architekturbewertung auf klassische Qualitätsmodelle der Bewertung von Software zurückgreift.

[243] Vgl. Kapitel 2.1.

[244] Vgl. Kapitel 2.2.5.1.

[245] Vgl. Kapitel 2.2.3.

[246] Vgl. Hansen /Wirtschaftsinformatik/ S. 163-165.

[247] Vgl. Raasch /Systementwicklung/ S. 32-35. Leider betrachtet Raasch jedoch die Auswirkungen einer Softwarearchitektur nicht durchgängig in seinem Qualitätsmodell, sondern schwerpunktmäßig bei den Qualitätsattributen Übertragbarkeit und Portabilität.

[248] Der Standard 9126 der ISO wird international verwendet und durch die ISO gepflegt.

sind gut verstanden.[250] Gegenwärtig kann angenommen werden, dass alle relevanten Qualitätsattribute zur Qualitätsbewertung von Software identifiziert sind. Im Folgenden werden diese hier verwendeten Qualitätsattribute aus dieser Perspektive durch die vertretene Sichtweise ‚technisch-fachliche Qualitätsattribute' genannt. Diese sind zur Bewertung einer Softwarearchitektur wichtig, weil sowohl die technische als auch die fachliche Seite (im Sinne der Fachlichkeit der Softwareentwicklung) einer Software und Softwareentwicklung direkt durch eine Softwarearchitektur beeinflusst wird.[251] In Anbetracht dessen, dass die zu erstellende Unternehmens-IT eines Unternehmens unterstützen soll,[252] kann unterstellt werden, dass zumindest ein indirekter, ggf. langfristig wirkender Einfluss der Unternehmens-IT auf den Erfolg eines Unternehmens besteht.

Dem entgegen existieren Qualitätsattribute, die noch nicht durchdringend erforscht und wenig verstanden sind und im Folgenden ‚geschäftliche Qualitätsattribute' genannt werden. Diese Qualitätsattribute erhalten ihre Relevanz zur Bewertung einer Softwarearchitektur einerseits durch eine enge Einbindung von Softwaresystemen in Unternehmen und weil dem Verständnis dieser Arbeit folgend eine SOA eine im Gesamtunternehmen eingesetzte Softwarearchitektur darstellt;[253] andererseits erhalten sie ihre Relevanz durch eine entsprechend große Tragweite der durch Softwarearchitekturen vorgegebenen Gestaltungsrichtlinien auf die Gestaltung (und somit auch Bewertung) einer Unternehmens-IT. Geschäftliche Qualitätsattribute bilden somit die Anforderungen ab, die ein Unternehmen an seine Unternehmens-IT stellt. Zu geschäftlichen Qualitätsattributen existieren keine Standards, allerdings liegen Einzelarbeiten zu einzelnen

---

[249] Barry Boehm und James A. McCall sind anerkannte Wissenschaftler auf dem Gebiet der Wirtschaftsinformatik. Beide Wissenschaftler haben Qualitätsmodelle aufgestellt, die in der Praxis und Theorie Verwendung finden.

[250] Dies zeigt sich in der Anwendungshäufigkeit der entsprechenden Modelle. Vor allem das Qualitätsmodell der ISO wird, vermutlich durch den Status als internationaler Standard, intensiv genutzt.

[251] Um zwei Beispiele zu nennen: Modelle der Softwareentwicklung werden durch die verwendete Softwarearchitektur beeinflusst. Deutlich wird dies, wenn man objektorientierte Architekturen verwendet, weil hier objektorientierte Modelle herangezogen werden können, um Softwarekomponenten zu modellieren, wohingegen eine Softwareentwicklung nach funktionaler Vorgehensweise solche Modelle nicht einsetzen kann. Auch Entwicklungsvorgehen werden z. B. beim Projektmanagement durch Softwarearchitekturen beeinflusst. Dies wird deutlich, wenn man die organisatorischen Stellen betrachtet, welche für eine SOA geschaffen werden sollten (vgl. Kapitel 4.4), und für andere Softwarearchitekturen nicht benötigt werden.

[252] Vgl. Kapitel 2.2.5.1.

[253] Vgl. Kapitel 2.2.3.

Qualitätsanforderungen eines Unternehmens an seine Unternehmens-IT vor, die als Qualitätsattribute zur Bewertung einer Softwarearchitektur interpretiert werden können.

Die Herleitung der beiden Teilbereiche des hier verwendeten Qualitätsmodells beruht auf unterschiedlichen Ansätzen: Technisch-fachliche Qualitätsattribute können über Standards und Standardarbeiten zu diesem Thema erhoben werden. Geschäftliche Qualitätsattribute werden durch ein fokussiertes Literaturstudium hergeleitet. Durch diese Vorgehensweise kann jedoch nicht beansprucht werden, dass alle für die Architekturbewertung relevanten geschäftlichen Qualitätsattribute in dieser Arbeit untersucht werden. Das Vorgehen ist unkritisch, weil Qualitätsattribute zweier unterschiedlicher Bewertungsperspektiven, nämlich die technisch-fachliche und die geschäftliche, einzeln konsistent abgeleitet werden und in ihren Ausprägungen keine Überschneidungen aufweisen.

Ein weiteres Problem von Qualitätsmodellen ist die Gewichtung der einzelnen Qualitätsattribute. Asundi, Kazman und Klein schlagen vor, für jede Bewertung eines Qualitätsattributs auch eine Gewichtung zu erheben und entsprechend in die Gesamtbewertung einzubeziehen.[254] Für die Bewertung dieser Arbeit wird ebenso eine Gewichtung der Qualitätsattribute im Hinblick auf eine Architekturbewertung aufgestellt und in der Nutzenbewertung beachtet. Weil jedoch auf allgemeinem Niveau für eine Einstufung eines Gewichtes argumentiert wird, wird der Gewichtung eine dreistufige Ordinalskala (niedriges, neutrales und hohes Gewicht) zugrunde gelegt.

Die Ausführungen verdeutlichen, dass im Folgenden nicht versucht werden kann, alle Qualitätsmodelle und Qualitätsattribute aus Literaturarbeiten zur Software- und Softwarearchitekturbewertung auf einen gemeinsamen Nenner zu bringen. Vielmehr wird ein Qualitätsmodell auf Basis etablierter Qualitätsmodelle und einzelner Arbeiten zu Qualitätsattributen aufgestellt, welches zur Bewertung einer *Softwarearchitektur* angemessen ist.

3.1.2 Rahmen eines Qualitätsmodells zur Softwarearchitekturbewertung

Ein Qualitätsmodell besteht aus Qualitätsattributen, die sich durch nicht-funktionale Anforderungen, die zur Nutzenbewertung einer Softwarearchitektur verwendet werden

---

[254] Vgl. Asundi, Kazman, Klein /Economic Approach/ S. 14.

können, repräsentieren lassen.[255] Zu solchen Qualitätsattributen gehören dem ISO-Standard 9126 folgend z. B. Portabilität und Effizienz.

In einem Unternehmen existieren verschiedene funktionale sowie nicht-funktionale Anforderungen an die Unternehmens-IT. Diese umfasst Personen, Informationen, Technologien, geschäftliche Funktion und organisatorische Strukturen.[256] Die Bestandteile und Eigenschaften haben nicht nur Einfluss auf ein Unternehmen und dessen Arbeitsweisen, sondern auch auf die funktionalen sowie nicht-funktionalen Anforderungen an eine Unternehmens-IT. Es kommt nicht selten vor, dass nicht-funktionale Anforderungen entscheidender auf den Nutzen einer Software wirken als funktionale Anforderungen.[257] Sie können sogar K.o.-Kriterien für den Einsatz von Softwaresystemen darstellen.[258] Der Nutzen einer Software wird auch durch nicht-funktionale Anforderungen beeinflusst.[259] Da jede Software einer Softwarearchitektur zugrunde liegt, wird der Nutzen einer Softwarearchitektur nicht nur durch den Erfüllungsgrad funktionaler Anforderungen, sondern auch durch den Erfüllungsgrad nicht-funktionaler Anforderungen bestimmt.[260]

Weil die Relevanz funktionaler Anforderungen für die Bewertung einer Softwarearchitektur in Frage gestellt werden kann, ist eine Fokussierung auf nicht-funktionale Anforderungen vordringlich. Dies begründet sich dadurch, dass einerseits angenommen werden kann, dass funktionale Anforderungen auf Basis jeder Softwarearchitektur realisiert werden können. Andererseits beinhaltet eine Nutzenbewertung funktionaler Anforderungen das Problem, dass konkrete funktionale Anforderungen zur Bewertung betrachtet werden müssten. Dies erscheint für eine vom konkreten Anwendungsfall losgelöste Nutzenbewertung einer Softwarearchitektur nicht möglich, weil eine sehr große Menge funktionaler Anforderungen betrachtet und bewertet werden müsste.[261] Infolgedessen

---

[255] Vgl. IEEE /Architectual Description/ S. 7. Vgl. auch Kapitel 2.2.5.1.

[256] Vgl. Zachman /Framework/ S. 461-469.

[257] Vgl. Sommerville /Software Engineering/ S. 111.

[258] Vgl. Moro, Lehner /Reengineering/ S. 35; Sommerville /Software Engineering/ S. 111.

[259] Vgl. Sommerville /Software Engineering/ S. 109-114.

[260] Vgl. Bass, Clements, Kazman /Software architecture/ S. 264; Hofmeister, Nord, Soni /Software Architecture/ S. 4-5.

[261] Für jedes Anwendungsfeld, in dem Software eingesetzt wird, müsste für eine solche Untersuchung eine umfassende Zusammenstellung aller möglichen funktionalen Anforderungen erhoben werden. Selbst bei der Fokussierung auf die wichtigsten funktionalen Anforderungen (diese Festlegung ist schon problematisch, wie Vorgehensweisen wie z. B. Quality Funktion Deployment aufzeigen)

werden zur Aufstellung eines Qualitätsmodells nicht-funktionale Anforderungen untersucht.

Wie in Kapitel 3.1.1 dargestellt, wird das Qualitätsmodell dieser Arbeit über technisch-fachliche Qualitätsattribute hinaus um geschäftliche Qualitätsattribute ergänzt. Diese bilden weitere Anforderungen eines Unternehmens an seine Unternehmens-IT ab. Der Rahmen des Qualitätsmodells besteht also aus der Einteilung der Qualitätsattribute in technisch-fachliche und geschäftliche Qualitätsattribute (vgl. Abb. 3-1).

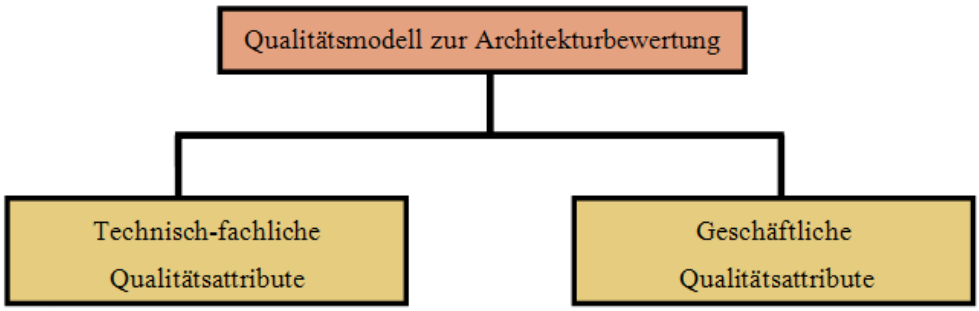

Abb. 3-1: Struktur des Qualitätsmodells

3.1.3 Technisch-fachliche Qualitätsattribute

Weil ein Qualitätsmodell aus vielen unterschiedlichen Perspektiven erstellt werden kann,[262] sind entsprechend viele technisch-fachliche Qualitätsmodelle mit dazugehörigen Qualitätsattributen von Autoren entwickelt worden.[263] Technisch-fachliche Qualitätsattribute wurden dementsprechend in der Literatur schon intensiv behandelt. Mit dem ISO-Standard ISO 9126:1991 wurde 1991 eine Version eines Standard-Qualitätsmodells entwickelt. Ein Jahrzehnt später, im Jahr 2001, wurde sie überarbeitet (ISO 9126-1:2001[264]). Die Version von 2001 bildet die Grundlage für die technisch-

---

würde vermutlich eine in dieser Arbeit nicht mehr zu bewältigende Menge an funktionalen Anforderungen zur Nutzenbewertung zu bewerten sein. Dies liegt daran, weil Software mittlerweile in sehr vielen unterschiedlichen Anwendungsfeldern eingesetzt wird.

[262] Vgl. Kapitel 3.1.1.

[263] Vgl. neben den hier verwendeten vier Qualitätsmodellen von Boehm, Raasch, McCall und der ISO auch Moro, Lehner /Reengineering/ S. 27-29; IEEE /Architectual Description/ S. 7; Bass, Clements, und Kazman stellen heraus, dass es notwendig zur Architekturbewertung ist, jeweils angepasste Qualitätsmodelle zu verwenden; vgl. Bass, Clements, Kazman /Software architecture/ S. 279-280.

[264] Vgl. ISO /9126/.

fachlichen Qualitätsattribute dieser Arbeit. Die Qualitätsmodelle der Autoren von Boehm (1978), Raasch (1993) und McCall (1994) werden in dieser Arbeit hinzugezogen.[265]

Qualitätsmodelle können nicht nur aus unterschiedlichen Perspektiven, sondern auch für verschiedene Bewertungsgegenstände erstellt werden.[266] Der ISO-Standard 9126 setzt den Fokus des Qualitätsmodells auf Software. Weil Anforderungen an Softwarearchitekturen (wie in Kapitel 3.1.1 dargestellt) davon verschieden seien können, werden einige Qualitätsattribute ergänzt bzw. so interpretiert, dass eine Bewertung einer Softwarearchitektur sinnvoll durchführbar ist. Eine solche Notwendigkeit zur Erweiterung und Änderung klassischer Qualitätsmodelle zur Softwarebewertung wurde auch durch andere Autoren erkannt und vorgenommen.[267]

Das Ergebnis der Erweiterung und Änderung der herangezogenen Qualitätsmodelle in Kapitel 3.1.3.1 ist, dass die bestehenden Qualitätsmodelle das Qualitätsmodell der ISO umfangreich unterstützen und dadurch die Korrektheit der Wahl des ISO-Standards 9126 als Ausgangspunkt des hier aufgestellten Qualitätsmodells aufgezeigt wird. Es werden einzelne Qualitätsattribute identifiziert, die im Qualitätsmodell der ISO keine Verwendung finden, jedoch für die Bewertung von Softwarearchitekturen sinnvoll und teilweise sogar notwendig erscheinen, um kein verzerrtes Bewertungsbild zu erzeugen. Dies sind z. B. die Qualitätsattribute Interoperabilität und Wiederverwendbarkeit. Einzelne Qualitätsattribute werden in der Definition erweitert interpretiert, damit diese für die Bewertung einer Softwarearchitektur genutzt werden können. Beispielsweise wird die Benutzbarkeit auf die ‚Benutzbarkeit aus Entwicklersicht' fokussiert.

---

[265] Vgl Boehm u. a. /Characteristics/, Raasch /Systementwicklung/ und McCall /Quality factors/.

[266] Qualitätsmodelle können alltäglich angetroffen werden: beim Einkaufen wird man eine Hose einem anderen Qualitätsmodell entsprechend bewerten (Aussehen, Farbe, Marke) als eine Zeitschrift (Art der Informationen, Sprache, Qualität der Recherche).

[267] Z. B. durch Bratthall u. a. sowie Moro und Lehner. Bratthall u. a. untersuchen spezielle Anwendungsfälle und erstellen kein allgemeingültiges Qualitätsmodell zur Softwarearchitekturbewertung. Vgl. Bratthall u. a. /Quality Requirements/ S. 2. Moro und Lehner stellen die Anforderungen an eine erstellte Softwarearchitektur vor und gehen dabei auf Qualitätsattribute ein, stellen jedoch nur dar, welche Bewertung im Projekt die entsprechenden Attribute erhielten und nicht, warum die Qualitätsattribute zur Softwarearchitekturentscheidung relevant waren. Moro, Lehner /Reengineering/ S. 27-29.

Während der Analyse der Qualitätsmodelle zeigt sich, dass die Autoren bzw. Gremien[268] in ihren Ausarbeitungen Unterschiede in der Aufstellung der jeweiligen Qualitäts- und Unterqualitätsattribute vornehmen. Beispielsweise wird ‚Testbarkeit' von Boehm und McCall als eigenständiges Qualitätsattribut aufgestellt, wohingegen die ISO und Raasch ‚Testbarkeit' als Unterqualitätsattribut der ‚Wartbarkeit' positionieren. Im Folgenden soll deswegen neben der Erläuterung der Qualitätsattribute eine kurze Reflexion der aufgestellten Qualitätsmodelle erfolgen. Weiterhin soll dargestellt werden, inwiefern die einzelnen Qualitätsattribute inkl. der Unterqualitätsattribute in dieser Arbeit Verwendung finden.

Die technisch-fachlichen Qualitäts- und Unterqualitätsattribute, die in dieser Arbeit Verwendung finden, werden im Folgenden erläutert (Kapitel 3.1.1.2 – 3.1.1.9). Bei der Definition der hier verwendeten Qualitätsattribute werden alle Unterqualitätsattribute aufgelistet und argumentiert, die in dieser Arbeit Verwendung finden. Zu den verwendeten Qualitätsattributen wird die Signifikanz argumentiert, die feststellt, weshalb diese Qualitätsattribute untersucht werden sollten und welche Gewichtung jedes Qualitätsattribut bei der Bewertung des Nutzenpotentials in dieser Arbeit erhält.[269]

3.1.3.1 Herleitung der technisch-fachlichen Qualitätsattribute

Die Grundlage der verwendeten Qualitätsattribute sind die des ISO-Standards 9126:2001.[270] Ergänzt werden diese durch einzelne Qualitätsattribute der Qualitätsmodelle der Autoren Boehm, Raasch und McCall. Die Qualitätsmodelle werden konsolidiert und ab Kapitel 3.1.3.2 auf ihre Eignung zur Bewertung einer Softwarearchitektur eingestuft.

---

[268] Im Folgenden nur noch als ‚Autoren' bezeichnet.

[269] Die Gewichtung kann bei der Anwendung in der Praxis durchaus unterschiedlich ausfallen, je nachdem, welche Schwerpunkte gesetzt werden. Beispielsweise würde die Gewichtung der Verlässlichkeit für Projekte im Bereich der Flugzeugsteuerung oder Auswertung von Wahlergebnissen wahrscheinlich höher gewichtet werden als bei einem Kassensystem in einer Gaststätte. Es wird zwar ärgerlich sein, wenn ein solches Kassensystem ausfällt, Abrechnungen (die Ehrlichkeit der Gäste und Kopfrechnenfähigkeiten vorausgesetzt) werden jedoch bis zur Wiederinbetriebnahme der Kasse durchgeführt werden können.

[270] Vgl. ISO /9126/ S. 7-11.

3.1.3.1.1 Qualitätsmodell der ISO (ISO 9126)

Im ISO-Standard 9126 werden implizit zwei Qualitätsmodelle aufgestellt: Das eine für ‚interne und externe Qualität' und das andere für ‚Qualität im Gebrauch'.[271] Die Qualitätsattribute des ISO-Standards 9126 für interne und externe Qualität sind:[272]

- Zweckmäßigkeit (Functionality)
- Zuverlässigkeit (Reliability)
- Benutzbarkeit (Usability)
- Effizienz (Efficiency)
- Wartbarkeit (Maintainability) und
- Portabilität (Portability)

Die Qualitätsattribute für Qualität im Gebrauch sind:[273]

- Effektivität (Effectivnes)
- Produktivität (Productivity)
- Sicherheit (Safety) und
- Zufriedenheit (Satisfaction)

Die Qualitätsattribute für ‚Qualität im Gebrauch' werden in dieser Arbeit zusammengefasst zum Qualitätsattribut ‚*Gebrauchsgerechtigkeit*'. Der ISO-Standard 9126 wurde 2001 gegenüber der Version von 1991 um diese Qualitätsattribute ergänzt, jedoch wurde kein hoher Detaillierungsgrad der einzelnen Qualitätsattribute ausgearbeitet, wie bei den Qualitätsattributen für ‚interne und externe Qualität'. Es wurden keine Unterqualitätsattribute für die einzelnen Qualitätsattribute der ‚Qualität im Gebrauch' aufgestellt. Das Qualitätsattribut ‚Gebrauchsgerechtigkeit' wird in dieser Arbeit im Weiteren, wegen des Bewertungsgegenstands der ‚Gebrauchsgerechtigkeit', nicht verwendet. Der Bewertungsgegenstand der Gebrauchsgerechtigkeit stellt Anforderungen an die Unternehmens-IT und bewertet, wie gut diese Anforderungen durch die Unternehmens-IT erfüllt werden. Der Hauptgegenstand der Anforderungen ist der Erfüllungsgrad *funktio-*

---

[271] Vgl. ISO /9126/ S. 3-5.
[272] Vgl. ISO /9126/ S. 7-11.
[273] Vgl. ISO /9126/ S. 12-13.

naler Anforderungen. Der Autor dieser Arbeit setzt voraus, dass funktionale Anforderungen durch jede Softwarearchitektur erfüllt werden können und dass sie für die Bewertung einzelner Softwarearchitekturen nicht zu einer unterschiedlichen Bewertung führen werden.[274] Neben dem Bewertungsgegenstand der ‚Gebrauchsgerechtigkeit', den funktionalen Anforderungen, können des Weiteren nicht-funktionale Anforderungen als Bewertungsgegenstand identifiziert werden. Nicht-funktionale Anforderungen werden durch die in dieser Arbeit aufgestellten technisch-fachlichen Qualitätsattribute abgedeckt und fließen somit explizit in die Bewertung mit ein. Aus diesem Grund wird die Gebrauchsgerechtigkeit nicht in das hier aufgestellte Qualitätsmodell zur Bewertung einer Softwarearchitektur hinzugezogen. Da einige Unterqualitätsattribute der Gebrauchsgerechtigkeit zu den untersuchten Qualitätsmodellen hinzuzählen, werden diese im Folgenden weiterhin vorgestellt, jedoch nicht im Qualitätsmodell dieser Arbeit verwendet.

3.1.3.1.2 Qualitätsmodell nach Boehm

Boehm identifiziert insgesamt sieben Qualitätsattribute:[275]

- Verständlichkeit (Understandability)
- Portabilität (Portability)
- Wartbarkeit (Maintainability)
- Testbarkeit (Testability)
- Benutzbarkeit (Usability)
- Zuverlässigkeit (Reliability)
- Effizienz (Efficiency)

Bis auf Testbarkeit und Verständlichkeit stimmt Boehm mit dem ISO-Standard 9126 überein.

---

[274] Vgl. Kapitel 3.1.2.

[275] Vgl. Boehm u. a. /Characteristics/ S. (3-23)-(3-26). Hier sind die Qualitätsattribute (in der Arbeit von Boehm „Software Characteristics" genannt) des „Revised Set" nach Boehm verwendet worden. Diese Überarbeitung durch Boehm ging aus Erkenntnissen hervor, dass sich einige Metriken der Qualitätsattribute überschneiden. Durch eine überarbeitete Menge an Qualitätsattributen ließ sich dieser Defekt beheben, vgl. Boehm u. a. /Characteristics/ S. (3-17)-(3-19).

In dieser Arbeit wird *Testbarkeit* als Unterqualitätsattribut des Qualitätsattributs Wartbarkeit aufgefasst. Testbarkeit nach Boehm ist die Förderung der Etablierung von Akzeptanzkriterien sowie die Unterstützung der Performanzmessung.[276] Die Testbarkeit kann der Wartbarkeit als Unterqualitätsattribut zugeordnet werden, weil das Qualitätsattribut Testbarkeit direkt im Zusammenhang mit der Erstellung und Wartung von Software zusammenhängt und nur in diesem Zusammenhang relevant ist.[277]

In dieser Arbeit wird *Verständlichkeit* nach Boehm als Unterqualitätsattribut des Qualitätsattributs Benutzbarkeit und Wartbarkeit aufgefasst. Verständlichkeit bezieht sich einerseits direkt auf Benutzbarkeit, wenn darunter die Verständlichkeit einer Anwendung für Anwender verstanden wird, und andererseits auf Wartbarkeit, wenn darunter die Verständlichkeit der Software (z. B. die Verständlichkeit der Softwarearchitektur oder des Softwarecodes) für Programmierer verstanden wird. Verständlichkeit als Unterqualitätsattribut wird folglich durch die Qualitätsattribute Benutzbarkeit und Wartbarkeit abgedeckt.[278]

3.1.3.1.3 Qualitätsmodell nach Raasch

Raasch identifiziert insgesamt neun Qualitätsattribute.[279] Er teilt diese ein in Qualitätsattribute aus Anwendersicht und Qualitätsattribute aus Entwicklersicht. Qualitätsattribute aus Anwendersicht sind:[280]

- Funktionserfüllung
- Effizienz
- Zuverlässigkeit
- Benutzbarkeit und

---

[276] Vgl. Boehm u. a. /Characteristics/ S. (3-11).

[277] Auch die ISO folgt dieser Einschätzung, indem das Qualitätsattribut Testbarkeit im Standard ISO 9126 ebenfalls der Wartbarkeit als untergeordnetes Qualitätsattribut zugeordnet wird, vgl. ISO /9126/ S. 11.

[278] Auch die ISO folgt der Einschätzung aus Anwendersicht, indem das Qualitätsattribut Verständlichkeit im Standard ISO 9126 ebenfalls der Benutzbarkeit als untergeordnetes Qualitätsattribut zugeordnet wird, vgl. ISO /9126/ S. 9.

[279] Vgl. Raasch /Systementwicklung/ S. 23-37. Raasch spricht in seinem Buch von „Qualitätsmerkmalen“.

[280] Vgl. Raasch /Systementwicklung/ S. 23-30.

- Sicherheit.

Bis auf das Qualitätsattribut *Sicherheit* stimmt Raasch mit dem ISO-Standard 9126 überein. Sicherheit nach Raasch bedeutet Sicherheit bezüglich unbefugter Nutzung, gefährlichen Zuständen, Datensicherheit und ordnungsgemäßer Verarbeitung. Diese Aspekte werden von der ISO unter den Qualitätsattributen Funktionalität, Verlässlichkeit und Gebrauchsgerechtigkeit (nach der Interpretation der ‚Quality in use' in dieser Arbeit) als Unterqualitätsattribute aufgefasst. In dieser Arbeit wird der Einteilung der ISO gefolgt und die Aspekte der Sicherheit unter diesen Qualitätsattributen geführt.[281]

Qualitätsattribute aus Entwicklersicht nach Raasch sind:[282]

- Wartbarkeit
- Portabilität
- Erweiterbarkeit und
- Wiederverwendbarkeit.

Bis auf die Qualitätsattribute Erweiterbarkeit und Wiederverwendbarkeit stimmt das Qualitätsmodell von Raasch mit dem ISO-Standard 9126 überein.

Die *Erweiterbarkeit* nach Raasch ist eine Aufgabe der Softwarewartung und eine Ausprägung der Änderbarkeit, weil jede Erweiterung einer Software immer auch eine Änderung dieser ist. Dieser Argumentation Raaschs folgend wird Erweiterbarkeit in dieser Arbeit dem Qualitätsattribut Wandlungsfähigkeit als Unterqualitätsattribut zugeordnet.

Das Qualitätsattribut *Wiederverwendbarkeit* wird als technisch-fachliches Qualitätsattribut im Qualitätsmodell dieser Arbeit gegenüber dem ISO-Standard 9126 hinzugefügt. Wiederverwendbarkeit wird gerade heute in Zeiten starken wirtschaftlichen Druckes als Hilfsmittel angesehen, Entwicklungszeit und -budget im Rahmen zu halten und die Qualität von Software zu verbessern.[283] Wiederverwendbarkeit sollte zu Beginn einer Softwareentwicklung geplant werden, weil sie nachträglich nur unbefriedigend zu erreichen ist.[284] Die Architektur einer Software hat durch den frühen Planungszeitpunkt einer Softwarearchitektur und durch den Charakter als Gestaltungsanleitung für eine Software

---

[281] Die ‚Gebrauchsgerechtigkeit' wird nicht in das Qualitätsmodell aufgenommen. Vgl. Kapitel 3.1.3.1.1.

[282] Vgl. Raasch /Systementwicklung/ S. 30-37.

[283] Vgl. Sommerville /Software Engineering/ S. 315-317.

direkten Einfluss auf die Wiederverwendbarkeit. Das Qualitätsattribut wird insofern als relevantes Qualitätsattribut zur Bewertung einer Softwarearchitektur angesehen.

3.1.3.1.4 Qualitätsmodell nach McCall

Die Qualitätsattribute nach McCall sind:[285]

- Korrektheit (Correctness)
- Zuverlässigkeit (Reliability)
- Effizienz (Efficiency)
- Integrität (Integrity)
- Benutzbarkeit (Usability)
- Wartbarkeit (Maintainability)
- Testbarkeit (Testability)
- Flexibilität (Flexibility)
- Portabilität (Portability)
- Wiederverwendbarkeit (Reuseability) und
- Interoperabilität (Interoperability)

Bis auf die Qualitätsattribute Korrektheit, Integrität, Testbarkeit und Interoperabilität stimmt das Qualitätsmodell von Raasch mit dem ISO-Standard 9126 überein.

Korrektheit nach McCall zielt auf Spezifikationserfüllung und Anforderungen der Anwender,[286] und entspricht dadurch dem Qualitätsattribut ‚Funktionalität' des ISO-Standards 9126.

Das Qualitätsattribut *Integrität* wird in dieser Arbeit dem Qualitätsattribut Gebrauchsgerechtigkeit gleichgesetzt, weil McCall unter Integrität die Kontrolle der Zugriffsmög-

---

[284] Vgl. Raasch /Systementwicklung/ S. 36.

[285] Vgl. McCall /Quality factors/ S. 961. McCall verwendet in seinem Beitrag den Begriff „Software Quality Factors". Allerdings wird schnell klar, dass er bei der Darstellung seines Qualitätsmodells den Begriff Qualitätsfaktor gleichbedeutend dem Begriff Qualitätsattribut, wie er bei anderen Autoren verwendet wird, einsetzt. Qualitätsattribute nach McCall sind beispielsweise „Access control" und „Access audit" beim ‚Quality factor' „Integrity", vgl. McCall /Quality factors/ S. 963.

[286] Vgl. McCall /Quality Factors/ S. 961.

lichkeiten durch Anwender versteht und diese Eigenschaft durch das Unterqualitätsattribut ‚Sicherheit' der Gebrauchsgerechtigkeit zugeordnet wurde.[287]

Das Qualitätsattribut *Testbarkeit* wird, wie bei Boehm argumentiert, der Wartbarkeit als Unterkriterium zugeordnet.[288]

Interoperabilität ist nach McCall ein Qualitätsattribut, im ISO-Standard 9126 dagegen ist es ein Unterqualitätsattribut der Funktionserfüllung. In dieser Arbeit wird die Auffassung der ISO vertreten und das Qualitätsattribut nach McCall als Unterqualitätsattribut der Funktionserfüllung zugeordnet. Dies begründet sich dadurch, weil die Fähigkeit einer Software, mit anderen Softwaresystemen interagieren zu können, für Software innerhalb eines Unternehmens heute als Voraussetzung angesehen werden kann. Die nicht-funktionale Anforderung Funktionserfüllung wird folglich um den Aspekt der Fähigkeit zur Interoperabilität erweitert.

3.1.3.1.5 Übersicht über die Qualitätsmodelle

In den vorhergegangenen Kapiteln wurden Qualitätsmodelle vorgestellt und dabei einzelne Qualitätsattribute identifiziert, die nicht mit dem ISO-Standard 9126 übereinstimmen (Tab. 3-1 fasst die Ergebnisse zusammen.) Entweder wurden diese als Qualitätsattribute identifiziert, die im ISO-Standard 9126 als Unterqualitätsattribute aufgeführt werden, oder sie sollten zur Bewertung einer Softwarearchitektur Beachtung finden.

Qualitätsattribut	*ISO 9126*	*Boehm*	*Raasch*	*McCall*
Benutzbarkeit	+	+	+	+
Effizienz	+	+	+	+
Wartbarkeit	+	+	+	+
Portabilität	+	+	+	+
Zuverlässigkeit	+	+	+	+

[287] Vgl. Kapitel 3.1.3.1.1.

[288] Vgl. Kapitel 3.1.3.1.2.

Qualitätsattribut	*ISO 9126*	*Boehm*	*Raasch*	*McCall*
Funktionserfüllung	+	–	+	–
Sicherheit	+	–	+	–
Effektivität	+	–	–	–
Produktivität	+	–	–	–
Zufriedenheit	+	–	–	–
Testbarkeit	–	+	–	+
Verständlichkeit	–	+	–	–
Wiederverwendbarkeit	–	–	+	+
Erweiterbarkeit	–	–	+	–
Integrität	–	–	–	+
Korrektheit	–	–	–	+
Flexibilität	–	–	–	+
Interoperabilität	–	–	–	+

Tab. 3-1: Technisch-fachliche Qualitätsattribute aus der Literatur

In Tab. 3-2 wird die in dieser Arbeit getroffene Aggregation dargestellt. Der in diesem Kapitel vorgestellte Teil des Qualitätsmodells, die technisch-fachlichen Qualitätsattribute, sind in Abb. 3-2 dargestellt.

Oey	*ISO 9126*	*Boehm*	*Raasch*	*McCall*
Funktions-erfüllung	Funktions-erfüllung	–	Funktions-erfüllung	Inter-operabilität
Wandlungs-fähigkeit	Wartbarkeit	Wartbarkeit Testbarkeit	Wartbarkeit Erweiterbarkeit	Flexibilität Wartbarkeit Testbarkeit
Benutzbarkeit	Benutzbarkeit	Benutzbarkeit	Benutzbarkeit	Benutzbarkeit

Oey	*ISO 9126*	*Boehm*	*Raasch*	*McCall*
		aus Anwender-sicht		
Verlässlichkeit	Verlässlichkeit	Verlässlichkeit	Verlässlichkeit[289]	Verlässlichkeit
Effizienz	Effizienz	Effizienz	Effizienz	Effizienz
Wieder-verwendbarkeit	–	Benutzbarkeit aus Ent-wicklersicht[290]	Wieder-verwendbarkeit	Wieder-verwendbarkeit
Portabilität	Portabilität	Portabilität	Portabilität	Portabilität
Gebrauchs-gerechtigkeit[291]	Sicherheit Effektivität Produktivität Zufriedenheit	Ver-ständlichkeit	Sicherheit	Korrektheit Integrität

Tab. 3-2: Aggregation technisch-fachlicher Qualitätsattribute

Im Folgenden werden die in dieser Arbeit verwendeten technisch-fachlichen Qualitätsattribute des Qualitätsmodells mit allen zugeordneten Unterqualitätsattributen vorgestellt und erläutert, weshalb diese zur Architekturbewertung relevant sind bzw. weshalb diese nicht in das Qualitätsmodell zur Bewertung einer Softwarearchitektur aufgenommen werden.

3.1.3.2 Funktionserfüllung

Das Qualitätsattribut Funktionserfüllung wird im ISO-Standard 9126 und von Raasch genannt.

---

[289] Raasch nutzt in seinen Ausführungen den Begriff „Zuverlässigkeit“. Der Begriff, wie er bei Raasch Verwendung findet, wird in diesem Beitrag jedoch als ‚Verlässlichkeit’ bezeichnet.

[290] Vgl. Boehm u. a. /Characteristics/ S. (3-12).

[291] Das Qualitätsattribut ist hier mit aufgeführt, damit deutlich wird, wie die Qualitätsmodelle der untersuchten Beiträge für diese Arbeit aggregiert wurden. Das Qualitätsattribut ‚Gebrauchsgerechtigkeit’ ist allerdings nicht Bestandteil des Qualitätsmodells dieser Arbeit. Vgl. Kapitel 3.1.3.1.

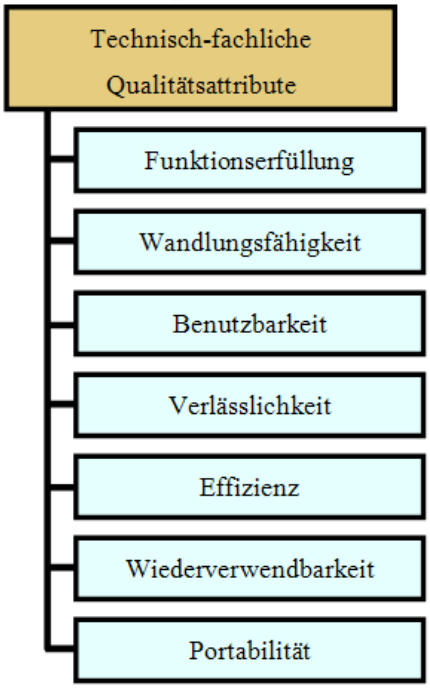

Abb. 3-2: Technisch-fachliche Qualitätsattribute

3.1.3.2.1 Erläuterung der Funktionserfüllung

ISO

Die ISO versteht unter der Funktionserfüllung die Fähigkeit eines Softwareproduktes, die Funktion zur Verfügung zu stellen, die explizit erwähnte und implizit erwartete Anforderungen erfüllt, wenn die Software unter festgelegten Bedingungen genutzt wird.[292] Die Unterqualitätsattribute der ISO für die Funktionserfüllung sind:[293] Angemessenheit (Suitability), Genauigkeit (Accuracy), Interoperabilität (Interoperability), Sicherheit (Security) und Gehorsamkeit (Compliance).

[292] Vgl. ISO /9126/ S. 7.

[293] Vgl. ISO /9126/ S. 7.

Raasch

Nach Raasch ist die Funktionserfüllung ein Maß für die Übereinstimmung zwischen geplantem und tatsächlich realisiertem Funktionsumfang.[294] Raasch stellt in seinen Ausführungen keine expliziten Unterqualitätsattribute auf.

Ein Qualitätsattribut nach Raasch, das der Funktionserfüllung teilweise zugeordnet werden kann, ist die *Sicherheit*. Sicherheit nach Raasch umfasst die Eigenschaften einer Software, die zu verhindern vermag, dass die Software (1) in einen gefährlichen Zustand gerät, (2) unbefugt genutzt wird, (3) Daten und Programme zerstört oder verfälscht werden können und dass (4) eine ordnungsgemäße und revisionsfähige Verarbeitung sichergestellt wird.[295] Sicherheitsmaßnahmen umfassen dabei auch das organisatorische Umfeld und die Hardware, auf der die Software ausgeführt wird. Hinsichtlich der Eigenschaften der Sicherheit, dass Software nicht unbefugt genutzt wird und dass Daten und Programme nicht zerstört oder verfälscht werden können wird der Sicherheit als Unterqualitätsattribut der Funktionserfüllung zugeordnet. Diese Auffassung entspricht auch dem Unterqualitätsattribut Sicherheit des ISO-Standards 9126. Die Aspekte, dass eine Software nicht in einen gefährlichen Zustand gerät und dass eine ordnungsgemäße und revisionsfähige Verarbeitung sichergestellt werden kann, werden der Funktionserfüllung zugeordnet.[296]

McCall

McCall stellt ein Qualitätsattribut ‚Funktionserfüllung' nicht explizit auf. Qualitätsattribute nach McCall, die der Funktionserfüllung als Unterqualitätsattribute zugeordnet werden können, sind Interoperabilität, Integrität und Korrektheit.

Unter *Interoperabilität* versteht McCall die Fähigkeit einer Software, mit einem oder mehreren anderen Softwaresystemen interagieren zu können. Dieses Verständnis entspricht dem Verständnis der ISO zum Unterqualitätsattribut Interoperabilität (des Qualitätsattributes Funktionserfüllung), weshalb Interoperabilität nach McCall als Unterqualitätsattribut zur Funktionserfüllung im Kontext dieser Arbeit verwendet wird.

---

[294] Vgl. Raasch /Systementwicklung/ S. 23.

[295] Vgl. Raasch /Systementwicklung/ S. 29.

[296] Vgl. Kapitel 3.1.3.2.2.

Integrität nach McCall ist die Fähigkeit einer Software, Zugriff auf die Funktionen der Software und gespeicherte Daten durch nicht berechtigte Personen zu kontrollieren.[297] Diese Definition entspricht der Definition der Sicherheit im ISO-Standard 9126, weshalb Integrität nach McCall und Sicherheit nach der ISO synonym behandelt werden.

Korrektheit nach McCall ist die Fähigkeit einer Software, die an sie gestellten Anforderungen zu erfüllen.[298] Diese Definition ist Teil der Definition der Funktionserfüllung gemäß der ISO, weshalb die Korrektheit nach McCall durch die Definition der Funktionserfüllung abgedeckt wird. Die Korrektheit als Qualitätsattribut nach McCall wird in dieser Arbeit folglich nicht explizit verwendet.

In Tab. 3-3 sind alle (Unter-)Qualitätsattribute der untersuchten Beiträge aufgelistet, die im Kontext dieser Arbeit der Funktionserfüllung zugeordnet werden können.

ISO 9126	*Boehm*	*Raasch*	*McCall*
Angemessenheit, Genauigkeit, Sicherheit, Gehorsamkeit, Interoperabilität	[Nennt das Qualitätsattribut nicht]	Keine explizite Nennung von Unterqualitätsattributen, jedoch wird *Sicherheit* als eigenständiges Qualitätsattribut aufgeführt.	Nennt das Qualitätsattribut nicht, jedoch werden *Interoperabilität (Modularity, Communications Commonality, Data commonality), Integrität (Access Control, Access Audit) und Korrektheit (Traceability, Consistency, Completeness)* als eigenständige Qualitätsattribute aufgeführt.

Tab. 3-3: Unterqualitätsattribute der Funktionserfüllung[299]

[297] Vgl. McCall /Quality factors/ S. 961.

[298] Vgl. McCall /Quality factors/ S. 961.

[299] Unterqualitätsattribute sind in Klammern hinter entsprechenden Qualitätsattributen gesetzt.

McCall stellt zu den Qualitätsattributen, die der Funktionserfüllung zugeordnet werden, weitere Unterqualitätsattribute vor. Diese Unterqualitätsattribute werden im Folgenden einzeln betrachtet und ihre Verwendung für das in dieser Arbeit zugrunde gelegte Qualitätsmodell dargestellt:

- Unterqualitätsattribute zu Interoperabilität sind Modularität (Modularity), gemeinsame Kommunikationsmechanismen (Communications commonality) und gemeinsame Datenmechanismen (Data commonality).[300] Modularität ist die Eigenschaft einer Software, einer Struktur voneinander unabhängiger Modulen zugrunde zu liegen.[301] Gemeinsame Kommunikationsmechanismen stellen die Fähigkeit einer Software dar, Kommunikation über Standardprotokolle und Standardschnittstellen zu ermöglichen.[302] Gemeinsame Datenmechanismen sind die Fähigkeit einer Software, Standards zur Datenrepräsentation anzubieten.[303] Das Qualitätsattribut Interoperabilität nach McCall ist durch diese Aufgliederung in Unterqualitätsattribute wesentlich detaillierter als die Definition der ISO. Um das Qualitätsmodell dieser Arbeit handhabbar zu gestalten, wird jedoch darauf verzichtet, Unterqualitätsattribute zu Unterqualitätsattributen zu definieren. Die von McCall angesprochenen Unterqualitätsattribute werden jedoch zur Bewertung des Unterqualitätsattributs ‚Interoperabilität' in dieser Arbeit herangezogen.
- Unterqualitätsattribute zu Integrität sind Zugriffskontrolle (Access Control) und Zugriffsprüfung (Access Audit).[304] Zugriffskontrolle ist die Fähigkeit einer Software, die die Kontrolle des Zugriffs auf die Software und die Daten ermöglicht.[305] Zugriffsprüfung ist die Fähigkeit einer Software, Festlegungen der Zugriffsberechtigungen auf Software und Daten ermöglichen zu können.[306] Wie schon festgestellt ist die Definition der Integrität nach McCall deckungsgleich mit der Definition der Sicherheit (als Unterqualitätsattribut der Funktionserfüllung) nach der ISO. Die Verfeinerung durch die Unterqualitätsattribute bei McCall ändert dies nicht, weshalb die

---

[300] Vgl. McCall /Quality factors/ S. 963.
[301] Vgl. McCall /Quality factors/ S. 965.
[302] Vgl. McCall /Quality factors/ S. 965.
[303] Vgl. McCall /Quality factors/ S. 965.
[304] Vgl. McCall /Quality factors/ S. 963.
[305] Vgl. McCall /Quality factors/ S. 965.
[306] Vgl. McCall /Quality factors/ S. 965.

explizite Betrachtung der Unterqualitätsattribute zu Integrität nach McCall in dieser Arbeit nicht erfolgt.

- Unterqualitätsattribute zu Korrektheit sind Nachweisbarkeit (Traceability), Konsistenz (Consistency) und Vollständigkeit (Completeness).[307] Nachweisbarkeit ist die Eigenschaft einer Software, unter gegebenen Entwicklungs- und Ausführungsumgebungen nachvollziehen zu können, welche und wie Anforderungen implementiert wurden.[308] Konsistenz ist die Eigenschaft einer Software, einheitliche Design- und Implementationsvorgehensweisen und –notationen zu verwenden.[309] Vollständigkeit ist die Eigenschaft einer Software, alle geforderten Funktionen implementiert zu haben.[310] Die Definition der Korrektheit nach McCall stimmt, wie schon erwähnt, mit der Definition der Funktionserfüllung nach der ISO überein. Nachweisbarkeit wird in dieser Arbeit als Bewertung der Hilfsmittel zur Bewertung der Funktionserfüllung angesehen und zur Bewertung einer Softwarearchitektur nicht verwendet.[311] Die Konsistenz nach McCall wird auch nicht als notwendiges Unterqualitätsattribut der Funktionserfüllung angesehen, weil die Konsistenz in dieser Definition (und aus der Perspektive, welcher die Betrachtung dieser Arbeit unterliegt: die der Software-Entwickler) die Verständlichkeit einer Software unterstützt. Die Verständlichkeit wird als Unterqualitätsattribut bei der Wandlungsfähigkeit bewertet werden und deswegen an dieser Stelle nicht als explizites Unterqualitätsattribut aufgenommen. Das Unterqualitätsattribut nach McCall ‚Vollständigkeit' ist in der Definition der Funktionserfüllung nach der ISO auch enthalten und wird ebenso nicht weiter explizit als Unterattribut aufgeführt.

Folgende Unterqualitätsattribute werden nicht in das Qualitätsmodell zur Softwarearchitekturbewertung aufgenommen:

- Angemessenheit
  Angemessenheit ist die Fähigkeit einer Software, Funktionen zur Verfügung zu stel-

---

[307] Vgl. McCall /Quality factors/ S. 963.

[308] Vgl. McCall /Quality factors/ S. 965.

[309] Vgl. McCall /Quality factors/ S. 965.

[310] Vgl. McCall /Quality factors/ S. 965.

[311] Obwohl an dieser Stelle erwähnt werden soll, dass eine SOA hier durchaus das Potential aufweist, ein geeignetes Hilfsmittel – das Service-Repository und das Service-Registry – zur Verfügung stellen zu können.

len, die die aufgestellten Anforderungen gemäß ihrer Spezifikationen erfüllen können.[312] Angemessenheit bezieht sich nur auf spezifizierte funktionale Anforderungen. Funktionale Anforderungen lassen sich allerdings auf Basis jeder Softwarearchitektur umsetzten. Dies begründet sich darin, dass Softwarearchitekturen eine Struktur vorgeben, welche Softwarekomponenten, die Beschreibung der extern sichtbaren Eigenschaften von Softwarekomponenten und die Beziehung der Softwarekomponenten untereinander beinhaltet. Eine solche strukturelle Vorgabe schränkt die Umsetzung funktionaler Anforderungen nicht ein. Im ungünstigsten Fall kann höchstens davon ausgegangen werden, dass zur Umsetzung einer funktionalen Anforderung extrem viel Aufwand betrieben werden muss, um diese umzusetzen. Eine solche Betrachtung kann im Rahmen dieser Arbeit jedoch nicht geschehen, weil ansonsten alle in der Theorie und Praxis auftretenden funktionalen Anforderungen im Hinblick auf ihre Schwierigkeit bei der Umsetzung in einer SOA betrachtet werden müssten. Dies ist sehr aufwändig, einerseits weil extrem viele funktionale Anforderungen an eine Unternehmens-IT gestellt werden können und andererseits, weil die konkrete Umsetzung einer SOA auf Basis unterschiedlicher Technologien geschehen kann, welches wiederum auch Einfluss auf den Aufwand zur Umsetzung einer funktionalen Anforderung hat.

- Genauigkeit

  Genauigkeit ist die Fähigkeit einer Software, die vorher festgelegten Ergebnisse in der gewünschten Genauigkeit zu liefern.[313] Die gewünschte Genauigkeit bezieht sich durch den Fokus auf Ergebnisse auf funktionale Anforderungen. Die gewünschte Genauigkeit funktionaler Anforderungen lässt sich durch eine entsprechend sorgfältige Implementation der entsprechenden Funktion erreichen. Eine Softwarearchitektur hat keinen Einfluss darauf, wie sorgfältig ein Software-Entwickler eine Funktion implementiert. Somit ist die Genauigkeit unabhängig von der eingesetzten Softwarearchitektur und wird nicht zur Bewertung einer solchen verwendet.

Aus diesen Überlegungen leitet sich die Definition des folgenden Kapitels ab.

---

[312] Vgl. ISO /9126/ S. 8; Sommerville /Sofware Engineering/ S. 369.

[313] Vgl. ISO /9126/ S. 8.

3.1.3.2.2 Arbeitsdefinition Funktionserfüllung

Die Funktionserfüllung ist die Fähigkeit eines Softwareproduktes, die Funktion zur Verfügung zu stellen, die explizit aufgestellte und implizit erwartete Anforderungen erfüllt, wenn die Software unter festgelegten Bedingungen genutzt wird.[314]

Unterqualitätsattribute, die durch Softwarearchitekturen beeinflusst werden können,[315] sind:[316]

- Sicherheit
  Sicherheit ist die Fähigkeit einer Software, Informationen und Daten so zu schützen, damit nicht autorisierte Personen oder Systeme diese weder lesen noch ändern können, wohingegen ein Zugang zu Informationen und Daten autorisierten Personen oder Systemen erlaubt wird.[317]
- Interoperabilität
  Interoperabilität ist die Fähigkeit einer Software, mit einem oder mehreren anderen Softwaresystemen interagieren zu können.[318]
- Gehorsamkeit
  Funktionale Gehorsamkeit ist die Fähigkeit einer Software Standards, Vereinbarungen oder Regeln (z. B. der Gesetzgebung) einhalten zu können.[319]

3.1.3.2.3 Signifikanz der Funktionserfüllung

Die Gewichtung der Funktionserfüllung wird neutral bewertet. Aus zwei Gründen ist das Qualitätsattribut ‚Funktionserfüllung' zur Bewertung einer Softwarearchitektur als

---

[314] Vgl. ISO /9126/ S. 7; Raasch /Systementwicklung/ S. 23. Raaschs Definition ist in der Definition der ISO enthalten, weshalb in dieser Arbeit als Definition der Funktionserfüllung die Definition der ISO übernommen wird.

[315] Zur Argumentation vgl. Kapitel 3.1.3.2.3.

[316] Raasch stellt in seiner Arbeit keine expliziten Unterqualitätsattribute auf, weshalb keine explizite Zusammenführung der Unterqualitätsattribute stattfinden kann. Stattdessen werden Qualitätsattribute der Autoren Raasch und McCall als Unterqualitätsattribute aufgenommen.

[317] Vgl. ISO /9126/ S. 8; McCall /Quality factors/ S. 961 (Qualitätsattribut ‚Integrität'). Hierzu gehört auch der Aspekt nach Raasch, dass eine Software nicht in einen gefährlichen Zustand geraten kann. Vgl. Raasch /Systementwicklung/ S. 29;

[318] Vgl. McCall /Quality factors/ S. 961; ISO /9126/ S. 8; IEEE /Glossary/ S. 4.

relevant anzusehen: Zum einen, weil die Funktionserfüllung aus der Sicht des Benutzers ein entscheidendes Kriterium ist, zum anderen, weil Softwarearchitekturen Einfluss auf die Funktionserfüllung (vor allem nicht-funktionaler Anforderungen) haben.

Der erste Punkt, dass die Funktionserfüllung aus der Sicht des Benutzers ein entscheidendes Kriterium ist,[320] lässt sich dadurch erklären, dass eine Software, die Anforderungen nicht erfüllt (unabhängig davon, ob diese explizit oder implizit aufgestellt wurden), nicht in der vorgesehenen Art und Weise verwendet werden kann. Dies kann sogar soweit führen, dass eine Software, die einzelne Funktionen nicht erfüllt, überhaupt nicht verwendet werden kann.[321] Eine solche Auswirkung kann sich schon bei einzelnen Unterqualitätsattributen ergeben. Sind Funktionen nicht angemessen umgesetzt, entsprechen sie z. B. nicht der geforderten Genauigkeit oder Sicherheit, oder folgen in nicht ausreichendem Maße den extern geforderten Gesetzen (Gehorsamkeit), ist die Funktionserfüllung und infolgedessen die Verwendbarkeit einer Software gefährdet.

Der zweite Punkt, dass Softwarearchitekturen Einfluss auf die Funktionserfüllung haben, insbesondere auf nicht-funktionale Anforderungen, erklärt sich dadurch, dass Softwarearchitekturen den Gestaltungsrahmen einer Software vorgeben: Softwarearchitekturen legen fest, wie Softwarekomponenten gestaltet werden sollen, welche Eigenschaften diese nach außen repräsentieren und wie die Beziehung zueinander gestaltet ist.[322] Nicht-funktionale Anforderungen werden durch die Gestaltungsmöglichkeiten einer Software beeinflusst. Dagegen hängt eine Softwarearchitektur i. d. R. nicht von den Details der fachlichen Anforderungen ab.[323]

Signifikanz der Unterqualitätsattribute

- Sicherheit

  Sicherheit im Sinne der Funktionserfüllung wird deswegen durch Softwarearchitekturen beeinflusst, weil eine Softwarearchitektur festlegt, welche Eigenschaften einer

---

[319] Vgl. ISO /9126/ S. 8. Hierzu gehört z. B. der Aspekt nach Raasch, dass eine Software die ordnungsgemäße und revisionsfähige Verarbeitung sicherstellen kann. Vgl. Raasch /Systementwicklung/ S. 29.

[320] Vgl. Raasch /Systementwicklung/ S. 23.

[321] Vgl. Mellis /Projektmanagement/ S. 65-66.

[322] Vgl. Kapitel 2.1.

[323] Vgl. Mellis /Projektmanagement/ S. 87.

Softwarekomponente nach außen sichtbar sind und wie Softwarekomponenten miteinander in Beziehung stehen. Sowohl der Grad der Preisgabe von Informationen nach extern als auch die Gestaltung der Beziehung zwischen Softwarekomponenten beeinflussen die Möglichkeiten, dass nicht autorisierte Personen Zugriff erlangen können. Die Gestaltung der Beziehung zwischen Softwarekomponenten beeinflusst darüber hinaus die Möglichkeit, wie Zugangsberechtigungen berechtigter Personen und Systeme kommuniziert werden können.

- Interoperabilität

  Softwarearchitekturen legen wie eben beschrieben fest, welche Eigenschaften extern sichtbar sind und wie Beziehungen zwischen Softwarekomponenten gestaltet sind. Um Interoperabilität zwischen Softwaresystemen erreichen zu können, müssen Informationen extern (zum anderen Softwaresystem) kommuniziert werden. Um interagieren zu können muss eine Verbindung zum anderen Softwaresystem aufgebaut werden. Eine solche Verbindung wird (1) durch die Beziehung von (2) Softwarekomponenten untereinander und zur Umgebung beeinflusst. Dies wird durch eine Softwarearchitektur mittels Gestaltungsrichtlinien festgelegt. Folglich wird die Fähigkeit zur Interoperabilität direkt durch Softwarearchitekturen beeinflusst. Darüber hinaus steigt die Relevanz der Interoperabilität, weil Informationsansprüche heute stark gestiegen sind und weil nicht nur Kommunikation mit bestehenden Geschäftspartnern stattfindet, sondern auch mit potentiellen Partnern/Kunden, zu denen (noch) keine absatzmarktbezogenen Beziehungen bestehen.[324] Durch ein gestiegenes Bedürfnis nach Information und nach deren Aktualität müssen Informationsansprüche heute häufiger, schneller und zielgruppenspezifischer bereitgestellt werden.[325] Die Entwicklung von Kooperationsfähigkeit erscheint auch immer wichtiger,[326] weil das Informationszeitalter durch Wertschöpfungsnetzwerke, die Kundenprozesse ganzheitlich unterstützen, geprägt ist.[327] Generell kann im Unternehmen auch das strate-

---

[324] Vgl. Werder, Grundei, Talaulicar /Unternehmenskommunikation/ S. 397-398.

[325] Vgl. Werder, Grundei, Talaulicar /Unternehmenskommunikation/ S. 397-398.

[326] Vgl. Lewin, Hunter /Information Technology/ S. 272-273.

[327] Vgl. Österle, Fleisch, Felt /Business Networking/ S. 3-4; Baumöl, Winter /Qualifikation/ S. 47. Als Beispiele können Bestell- und Lieferprozesse bei Online-Versandhäusern dargestellt werden: Ein Kunde kann schon heute Bestellungen bei einem Lieferanten aufgeben und bekommt nicht mehr mit, dass im Hintergrund in Echtzeit Bonitätsprüfungen beim Kreditgeber, Lagerbestandsabfragen bei Subunternehmern sowie Lieferzeiten und –bedingungen bei Versanddienstleistern stattfinden. Ein weiteres Beispiel sind Reisebüros: Hier wird in Echtzeit die Belegung von Hotelzimmern, Ver-

gische Ziel verfolgt werden, eine Verbesserung der Außenbeziehungen durch den Einsatz der Unternehmens-IT zu erreichen.[328] Um diesem Ziel nachgehen zu können ist die Fähigkeit der Unternehmens-IT wichtig, sich zu externen Systemen verbinden und mit diesen zusammenarbeiten zu können. Daraus ergibt sich, dass eine Unternehmens-IT heute und zukünftig verstärkt mit anderen und in zunehmendem Maße unternehmensexternen Systemen kommunizieren muss.

- Gehorsamkeit

  Funktionale Gehorsamkeit ist die Fähigkeit einer Software Standards, Vereinbarungen oder Regeln (z. B. der Gesetzgebung) einhalten zu können.[329] Die Gehorsamkeit einer Software in Hinblick auf funktionale Anforderungen bedeutet die Umsetzung von Funktionen, die sich an Standards, Vereinbarungen oder Regeln halten. Dies wird durch die Software-Entwickler erreicht, indem diese die entsprechenden Standards, Vereinbarungen oder Regeln bei der Implementation von Funktionen berücksichtigen und die Software so programmieren, dass die Standards, Vereinbarungen und Regeln eingehalten werden. Funktionale Gehorsamkeit ist somit zum einen eine direkte Folge von genauer Umsetzung funktionaler Anforderungen. Sowohl zur Genauigkeit als auch zur Umsetzung funktionaler Anforderungen wurde oben argumentiert, dass diese nicht durch Softwarearchitekturen beeinflusst werden. Durch ihre Wirkung auf die Gestaltung der Beziehungen zwischen Softwarekomponenten hat eine Softwarearchitektur Einfluss darauf, wie Standards, Vereinbarungen oder Regeln der Kommunikation und Datenübermittlung unterstützt werden.

Einstufung der Gewichtung des Qualitätsattributes

Die Funktionserfüllung ist generell relevant zur Bewertung von Softwarearchitekturen. Weil nicht-funktionale Anforderungen jedoch im Qualitätsmodell dieser Arbeit explizit bewertet werden und die qualitative Bewertung funktionaler Anforderungen zur Bewer-

---

fügbarkeit von Flugrouten und die Belegung von Sitzplätzen für bestimmte Routen mit dazugehörigen Tarifen sowie der gleichzeitige Abschluss von Versicherungen zur Reise angeboten. In allen Fällen sind mehrere Unternehmen mit ihrem entsprechenden Leistungsangebot in den Geschäftsprozess eines Dienstleisters eingebunden.

[328] Vgl. Macharzina /Unternehmensführung/ S. 654-657; Gizanis, Legner, Österle /Kooperative Auftragsabwicklung/ S. 48.

[329] Vgl. ISO /9126/ S. 8.

tung einer Softwarearchitektur nicht relevant ist, [330] wird die Gewichtung der Funktionserfüllung niedrig eingestuft.

3.1.3.3 Wandlungsfähigkeit

Das Qualitätsattribut ‚Wandlungsfähigkeit' wird unter dieser Bezeichnung weder von der ISO noch von Boehm, Raasch oder McCall genannt. In dieser Arbeit wird ‚Wandlungsfähigkeit' anstelle des Begriffes ‚Wartbarkeit' verwendet. Damit soll ausgedrückt werden, dass ‚Wandlungsfähigkeit' über den grundlegenden Begriff der ‚Wartung' hinaus explizit die Änderungen beinhaltet, die nicht nur der Fehlerbehebung dienen. Zwar teilt das IEEE diese Auffassung bereits für den Begriff der Wartung,[331] jedoch verstehen einzelne Autoren, beispielsweise McCall[332], unter der Wartung nur die Fehlerbehebung einer Software.

Zu dem Begriff Wandlungsfähigkeit im Sinne dieser Arbeit werden jedoch von allen vier für diese Untersuchung herangezogenen Autoren Qualitätsattribute unter der Bezeichnung ‚Wartbarkeit' verwendet. Das Qualitätsattribut ‚Wartbarkeit' wird demnach von allen Autoren übernommen. Darüber hinaus werden im Folgenden weitere Qualitätsattribute der Autoren inklusive der entsprechenden Unterqualitätsattribute der Wandlungsfähigkeit zugeordnet. Dies sind unter anderem die Qualitätsattribute Erweiterbarkeit (nach Raasch) und Testbarkeit (nach McCall).

3.1.3.3.1 Erläuterung der Wandlungsfähigkeit

Der Wandel von Unternehmen kann nach Wandlungsbedarf, Wandlungsbereitschaft und der Wandlungsfähigkeit differenziert werden.[333] Wandlungsbedarf ist das Ausmaß sachlich notwendiger Veränderungen des Unternehmens bzw. seiner Subsysteme und

---

[330] Vgl. Kapitel 3.1.3.2.1.

[331] Maintenance: „(1) The process of modifying a software system or component after delivery to correct faults, improve performance or other attributes, or adapt to a changed environment. (2) The process of retaining a hardware system or component in, or restoring it to, a state in which it can perform its required functions." Vgl. IEEE /Glossary/ S. 47.

[332] Vgl. McCall /Quality factors/ S. 961.

[333] Vgl. zum folgenden Absatz Krüger /Management/ S. 228-236.

kann durch systeminterne oder -externe Impulse ausgelöst werden.[334] Wandlungsbereitschaft ist die subjektive Bereitschaft bzw. Akzeptanz des Wandels. Sie stellt die Einstellung der Beteiligten dar und kann nicht direkt durch eine Software oder Softwarearchitektur festgelegt werden. Wandlungsfähigkeit ist die Beherrschung von Wandlungsprozessen und wird somit direkt durch das Objekt (in diesem Fall durch die Unternehmens-IT), das dem Wandlungsbedarf ausgesetzt ist, beeinflusst. Die Wandlungsfähigkeit lässt sich auf verschiedenen Ebenen betrachten: Auf einer Individualebene, auf einer Ebene der organisatorischen Einheit und auf einer Unternehmensebene. Die Wandlungsfähigkeit einer Unternehmens-IT kann auf diesen Ebenen unterschiedlich ausgeprägt sein. Primär soll die Wandlungsfähigkeit von Informationssystemen, die auf Basis von Softwarearchitekturen erstellt werden, sicherstellen, dass auf Änderungen innerhalb von Organisationen pro-aktiv reagiert werden kann.[335] Insofern ist insbesondere die Ebene des Unternehmens neben der Ebene der Organisationseinheiten von besonderer Bedeutung für die Bewertung einer Softwarearchitektur.

Anstelle der Wandlungsfähigkeit wird in der Literatur von Flexibilität oder Evolvierbarkeit gesprochen.[336] Flexibilität wird vom IEEE definiert als die Möglichkeit, eine Komponente in Anwendungen oder Umgebungen einzusetzen, für die sie nicht explizit spezifiziert wurde.[337] Der Begriff ‚Flexibilität' nimmt folgenden Bezug zur Wandlungsfähigkeit ein: Wandlungsbedarf beinhaltet eine sachliche und eine zeitliche Dimension.[338] Wenn einem Wandlungsbedarf ein in angemessener Zeit aktivierbares Wandlungspotential gegenübersteht, ist ein System flexibel.[339] Flexibilität umfasst für ein Unternehmen auch die Schaffung von Chancen durch Innovationskraft.[340] In diesem Kontext wird von Wandlungsfähigkeit auch als Produktivität zweiter Ordnung gesprochen.[341] Die besondere Herausforderung liegt hier in der Änderung bestehender Struktu-

---

[334] Vgl. Sommerville /Software Engineering/ S. 389; Aier, Schönherr /Felxibilisierung/ S. 8; Oxman, Smith /Structual Change/ S. 82.

[335] Vgl. Gronau /Wandlungsfähige Informationssystemarchitekturen/ S. 41-44.

[336] Vgl. beispielsweise Moro, Lehner /Reengineering/ S. 27-29; Stiemerling /Web-Services/ S. 436-437.

[337] IEEE /Glossary/ S. 31.

[338] Vgl. Gronau /Wandlungsfähige Informationssystemarchitekturen/ S. 206.

[339] Vgl. Gronau /Wandlungsfähige Informationssystemarchitekturen/ S. 205-206; Kieser, Walgenbach /Organisation/ S. 425; Aier, Schönherr /Flexibilisierung/ S. 9.

[340] Vgl. Aier, Schönherr /Flexibilisierung/ S. 9.

[341] Vgl. Hill, Fehlbaum, Ulrich /Organisationslehre/ S. 164-166.

ren, Prozesse und IT-Systeme, nicht in der abgestimmten Einführung dieser.[342] Aktuell ist die Flexibilität einer Unternehmens-IT neben der Erweiterbarkeit eine häufig genannte nicht-funktionale Anforderung an Softwaresysteme.[343] Aier und Schönherr schreiben sogar: „Als globales Ziel kann nachhaltige Flexibilisierung der IT-Infrastruktur und damit die nachhaltige Flexibilisierung der Unternehmensprozesse angesehen werden.“[344] Evolvierbarkeit bezeichnet die Anpassbarkeit an sich ändernde Anforderungen, um schnell auf Veränderungen am Markt reagieren zu können.[345] Dieser Begriff der Evolvierbarkeit ist synonym zum Begriff Flexibilität und wird im Folgenden nicht weiter explizit von der Flexibilität unterschieden.

ISO

Wie oben erwähnt ist Wandlungsfähigkeit nicht explizit im ISO-Standard 9126 aufgeführt. Jedoch wird im ISO-Standard 9126 ‚Wartbarkeit' mit den zugeordneten Unterqualitätsattributen vorgestellt. Dem ISO-Standard 9126 nach bedeutet ‚Wartbarkeit'[346] die Fähigkeit eines Softwareproduktes, modifiziert werden zu können.[347] Änderungen können Korrekturen, Verbesserungen oder Anpassungen der Software an geänderte Umweltbedingungen, Anforderungen und funktionale Spezifikationen darstellen. Unterqualitätsattribute des ISO-Standards 9126 sind:[348] Analysierbarkeit (Analysability), Änderbarkeit (Changeability), Stabilität (Stability), Testbarkeit (Testability) und Gehorsamkeit (Compliance).

Boehm

Boehm versteht unter der Wartbarkeit die Fähigkeit einer Software geändert werden zu können. Aktualisierungen werden mit dem Ziel verfolgt, neue Anforderungen hinzufü-

---

[342] Vgl. Aier, Schönherr /Flexibilisierung/ S. 11.

[343] Vgl. Moro, Lehner /Reengineering/ S. 25-26; Aier, Schönherr /Flexibilisierung/ S. 15.

[344] Aier, Schönherr /Flexibilisierung/ S. 15.

[345] Strabel /Neue Architektur/ S. 89; Jung /IT-Architekturmodell/ S. 313.

[346] Das IEEE unterscheidet unterschiedliche Arten der Wartung: adaptive, korrektive, perfektionistische und präventive Wartung, vgl. IEEE /Glossary/ S. 3 & S. 18. Eine solche Unterscheidung ist für die scope dieser Arbeit nicht notwendig, da es hier um die generelle Fähigkeit der Wartung geht. Welches Ziel mit einer Wartung verfolgt wird – ob es somit eine korrektive oder perfektionistische Wartung darstellt – ist zur Bewertung Wandlungsfähigkeit irrelevant.

[347] Vgl. zum folgenden Absatz ISO /9126/ S. 10.

[348] Vgl. ISO /9126/ S. 7.

gen zu können.[349] Unterqualitätsattribute der Wartbarkeit sind nach Boehm Verständlichkeit, Testbarkeit und Änderbarkeit.

Boehm stellt die Testbarkeit trotz der Zuordnung zur Wartbarkeit allerdings zusätzlich als eigenständiges Qualitätsattribut auf. Wie in Kapitel 3.1.3.1.2 dargelegt, wird dieser Ansicht nicht gefolgt, sondern sowohl das Qualitätsattribut ‚Testbarkeit' nach Boehm als auch das Unterqualitätsattribut ‚Testbarkeit' nach Boehm in dieser Arbeit als Unterqualitätsattribut der Wandlungsfähigkeit angesehen.

Raasch

Dem Qualitätsattribut ‚Wandlungsfähigkeit' werden in dieser Arbeit die Qualitätsattribute ‚Wartbarkeit' und ‚Erweiterbarkeit' nach Raasch zugeordnet.[350] Wartbarkeit nach Raasch bezieht sich auf die Zeitdauer, die erforderlich ist, um einen Fehler zu beheben. Als Unterqualitätsattribute der Wartbarkeit identifiziert er Verständlichkeit, Änderbarkeit und Testbarkeit.[351] Erweiterbarkeit nach Raasch verfolgt das Ziel, zusätzliche Funktionen auch noch nach Fertigstellung einer Software hinzufügen zu können.[352]

McCall

Wandlungsfähigkeit im Sinne dieser Arbeit wird bei McCall durch die Qualitätsattribute Wartbarkeit, Flexibilität und Testbarkeit abgedeckt.[353] Wartbarkeit ist die Fähigkeit, Fehler einer in Benutzung befindlichen Software mit möglichst geringem Aufwand finden und beheben zu können.[354] Flexibilität ist die Fähigkeit einer in Benutzung befindlichen Software, mit möglichst geringem Aufwand geändert werden zu können.[355] Testbarkeit ist die Fähigkeit einer Software, diese mit möglichst geringem Aufwand testen zu können um sicherzustellen, dass sie ihre vorgesehene Funktion erfüllt.[356]

---

[349] Vgl. Boehm u. a. /Characteristics/ S. (3-10).
[350] Vgl. Kapitel 3.1.3.1.3.
[351] Vgl. Raasch /Systementwicklung/ S. 31-32.
[352] Vgl. Raasch /Systementwicklung/ S. 30-31.
[353] Vgl. McCall /Quality factors/ S. 963.
[354] Vgl. McCall /Quality factors/ S. 961.
[355] Vgl. McCall /Quality factors/ S. 961.
[356] Vgl. McCall /Quality factors/ S. 961.

Zu den oben vorgestellten Qualitätsattributen identifiziert McCall die Unterqualitätsattribute Konsistenz (Consistency), Einfachheit (Simplicity), Prägnanz (Conciseness), Modularität (Modularity), Selbstbeschreibung (Self-descriptiveness), Allgemeingültigkeit (Generality), Erweiterbarkeit (Expandability) und Ausrüstung (Instrumentation). McCall trifft keine eindeutige Zuordnung, sondern ordnet diese einzelnen Unterqualitätsattribute mehreren der oben vorgestellten Qualitätsattributen nach McCall zu.[357] Dies wird als Indiz aufgefasst, dass eine Zusammenfassung der drei Qualitätsattribute inklusive der Unterqualitätsattribute zur ‚Wandlungsfähigkeit' im Sinne dieser Arbeit sinnvoll ist. Die Unterqualitätsattribute nach McCall können folglich als Unterqualitätsattribute der Wandlungsfähigkeit verwendet werden.

Tab. 3-4 fasst alle Unterqualitätsattribute der Wandlungsfähigkeit, die von den untersuchten Beiträgen aufgestellt wurden, zusammen.

Aus der vollständigen Auflistung aller Unterqualitätsattribute (vgl. Tab. 3-4) werden die folgenden Qualitätsattribute nicht in das Qualitätsmodell zur Architekturbewertung dieser Arbeit übernommen:

- Ausrüstung

  Ausrüstung nach McCall dient der Erfassung von Testwerten[358] und wird als Bestandteil der Testbarkeit aufgefasst.

- Flexibilität

  McCall stellt das Qualitätsattribut Flexibilität separat auf, um in Abgrenzung zu seinem Verständnis der Wartbarkeit Fähigkeiten der Erweiterbarkeit in seinem Qualitätsmodell abbilden zu können. McCall definiert Wartbarkeit nämlich nur in Bezug auf Fehlerbehebung. In dieser Arbeit jedoch werden darüber hinausgehend unter der Wandlungsfähigkeit alle auftretenden Änderungen aufgefasst. Aufgrund dessen wird das Qualitätsattribut Flexibilität nach McCall zum einen nicht explizit als Unterqualitätsattribut der Wandlungsfähigkeit aufgenommen, weil es durch das Qualitätsattribut Wandlungsfähigkeit dieser Arbeit abgedeckt wird; zum anderen wird Flexibilität nach McCall im Kontext dieser Arbeit als ein Ergebnis hoher Wandlungsfähigkeit interpretiert und nicht als Unterqualitätsattribut aufgenommen.

---

[357] Beispielsweise wird bei McCall das Unterqualitätsattribut ‚Selbst-Beschreibung' sowohl dem Qualitätsattribut ‚Flexibilität' als auch dem Qualitätsattribut ‚Testbarkeit' zugeordnet.

[358] Vgl. McCall /Quality factors/ S. 965.

ISO 9126	*Boehm*	*Raasch*	*McCall*
Wartbarkeit (Analysierbarkeit, Änderbarkeit, Stabilität, Testbarkeit, Gehorsamkeit),	Wartbarkeit (Verständlichkeit, Änderbarkeit, Testbarkeit)	Wartbarkeit (Verständlichkeit, Änderbarkeit, Testbarkeit), Erweiterbarkeit	Wartbarkeit, Flexibilität, Testbarkeit (Konsistenz, Einfachheit, Prägnanz, Modularität, Selbstbeschreibung, Allgemeingültigkeit, Erweiterbarkeit, Ausrüstung)[359]

Tab. 3-4: Unterqualitätsattribute der Wandlungsfähigkeit[360]

- Verständlichkeit

  Verständlichkeit bedeutet, dass der Zweck einer Software bzw. eines Softwareelements eindeutig klar gemacht werden kann.[361] Je kürzer die Zeit, die benötigt wird, um eine Software zu verstehen, desto verständlicher ist diese.[362] Verständlichkeit als Unterqualitätsattribut wird in dieser Arbeit nicht an dieser Stelle untersucht, sondern als Unterqualitätsattribut der Benutzbarkeit. Weil Benutzbarkeit in dieser Arbeit aus der Perspektive der Software-Entwickler betrachtet wird[363] und die Verständlichkeit einer Software direkte Auswirkungen auf die Benutzbarkeit aus Entwicklersicht hat,

---

[359] Da McCall einzelne der dargestellten Unterqualitätsattribute mehreren der drei Qualitätsattributen zuordnet, wird hier keine Unterscheidung der Zuordnung getroffen, sondern die Unterqualitätsattribute werden (nicht mehrfach) hintereinander aufgelistet.

[360] Wandlungsfähigkeit wurde von einem der Autoren als Qualitätsattribut so benannt. Hier wird der Begriff ‚Wartbarkeit' fast synonym zur Wandlungsfähigkeit interpretiert, weshalb in der Tabelle eine Unterscheidung zwischen Qualitätsattributen und Unterqualitätsattributen der einzelnen Autoren durch Klammersetzung angedeutet wird. Die Unterqualitätsattribute sind in Klammern hinter den entsprechenden Qualitätsattributen gesetzt. (Zu McCall siehe Text.)

[361] Vgl. Boehm u. a. /Characteristics/ S. (3-4)-(3-6); Raasch /Systementwicklung/ S. 31.

[362] Vgl. Raasch /Systementwicklung/ S. 31.

[363] Vgl. Kapitel 3.1.3.4.

würde eine Bewertung an dieser Stelle zu einer Doppelbewertung des Unterqualitätsattributes Verständlichkeit führen.

- Wartbarkeit
  Wartbarkeit wird synonym der Wandlungsfähigkeit angesehen.[364]
- Allgemeingültigkeit
  Allgemeingültigkeit nach McCall gibt die Breite der angebotenen Funktionen einer Software an.[365] Die Allgemeingültigkeit einer Software ist von einer verwendeten Softwarearchitektur unabhängig, weil sich Allgemeingültigkeit auf den gesamten Funktionsumfang einer Software bezieht. Der Funktionsumfang einer Software ist jedoch prinzipiell unabhängig von einer Softwarearchitektur, weil eine Softwarearchitektur Richtlinien zur Gestaltung einer Software vorschreibt und keine Einschränkungen in Bezug auf die Gesamtmenge angebotener Funktionen setzt.
- Gehorsamkeit
  Gehorsamkeit in Bezug auf Wandlungsfähigkeit bedeutet die Fähigkeit einer Software, Standards oder Vereinbarungen mit Bezug zur Wandlungsfähigkeit einhalten zu können.[366] Standards und Vereinbarungen beziehen sich darauf, wie eine Wandlung durchgeführt werden soll, z. B. wie ein Vorgehen zur Wandlung gestaltet ist. Das Einhalten von Standards und Vereinbarungen mit Bezug zur Wandlungsfähigkeit wird nicht durch eine Softwarearchitektur beeinflusst, weil eine Softwarearchitektur nicht vorschreibt, mit welchem Vorgehen eine Wandlung der Unternehmens-IT durchzuführen ist.

Aus diesen Überlegungen leitet sich die Definition des folgenden Kapitels ab.

3.1.3.3.2 Arbeitsdefinition Wandlungsfähigkeit

Wandlungsfähigkeit ist die Fähigkeit eines Softwareproduktes, verändert werden zu können. Änderungen können Korrekturen, Verbesserungen oder Anpassungen einer Software umfassen. Sie können durch geänderte Umweltbedingungen, Anforderungen oder funktionale Spezifikationen ausgelöst werden.

---

[364] Vgl. hierzu die Einleitung zum Kapitel 3.1.3.3.

[365] Vgl. McCall /Quality factors/ S. 965.

[366] Vgl. ISO /9126/ S. 8.

Unterqualitätsattribute werden von allen Autoren aufgestellt (siehe Tab. 3-4). In dieser Arbeit werden die Unterqualitätsattribute zur Wandlungsfähigkeit, die durch Softwarearchitekturen beeinflusst werden können,[367] folgendermaßen aufgestellt:

- Änderbarkeit
  Änderbarkeit bedeutet die Fähigkeit, spezifische Änderungen an einer Software vornehmen zu können.[368]
- Erweiterbarkeit
  Erweiterbarkeit ist die Fähigkeit, eine Software auch nach Fertigstellung um zusätzliche Anforderungen (meist funktionaler Art) oder um neue Datenansprüche (z. B. das Speichern vorher nicht erfasster Daten) erweitern zu können.[369]
- Testbarkeit
  Testbarkeit ist die Fähigkeit, eine Software nach der Erstellung bzw. Änderung validieren zu können.[370] Es wird hierbei festgestellt, ob zuvor festgelegte Kriterien bzw. Anforderungen erfüllt werden.[371]
- Stabilität (ISO)
  Stabilität einer Software stellt die Fähigkeit dar, unbeabsichtigte Auswirkungen durch Änderungen an der Software zu verhindern.[372]

3.1.3.3.3 Signifikanz der Wandlungsfähigkeit

Die Wandlungsfähigkeit einer Unternehmens-IT wird als hoch relevant für Unternehmen angesehen und geht deswegen mit einem hohen Gewichtungsfaktor in die Bewertung dieser Arbeit ein.

Das Management von Veränderungen eines Unternehmens, für die vielfältige Gründe existieren,[373] ist einer der wichtigsten Faktoren für den Erfolg eines Unternehmens.[374]

---

[367] Zur Argumentation vgl. Kapitel 3.1.3.3.3.

[368] Vgl. ISO /9126/ S. 10; Boehm und Raasch erläutern die Änderbarkeit nicht explizit.

[369] Vgl. Raasch /Systementwicklung/ S. 30-31; McCall /Quality factors/ S. 965; darüber hinaus vgl. IEEE /Glossary/ S. 30.

[370] Vgl. ISO /9126/ S. 11.

[371] Vgl. Boehm u. a. /Characteristics/ S. (3-11)-(3-12); McCall /Quality factors/ S. 961.

[372] Vgl. ISO /9126/ S. 10.

Dieses Management umfasst Information und Kommunikation,[375] folglich auch Informationssysteme und unternehmensweit betrachtet die gesamte Unternehmens-IT. Der Lebenszyklus einer Unternehmens-IT beträgt mehrere Jahre: Allein die Einführung neuer Software kann sich über einen Zeitraum von mehreren Monaten oder sogar Jahren erstrecken.[376] Anschließend wird Software i. d. R. mehrere Jahre und nicht selten Jahrzehnte betrieben.[377] Dementsprechend kann die Wandlungsfähigkeit in zwei zeitlichen Abschnitten im Lebenszyklus[378] einer Software betrachtet werden: Erstens der Zeitabschnitt der Entwicklung und zweitens der Zeitabschnitt der Anwendung und Wartung einer Software.

Der Anspruch der Wandlungsfähigkeit wird schon während des ersten Zeitabschnittes erhoben. Hier kann weiter zwischen individualentwickelter Software und Standardsoftware[379] unterschieden werden. Im Fall der Individualsoftwareentwicklung unterliegen die verschiedenen Gegenstände einer Software wie die Dokumentation, der Quellcode, die Schnittstellen, die Softwarearchitektur und erstellte Modelle verschieden stark ausgeprägten Änderungen.[380] Eine Dynamik von Anforderungen ist bei fast jeder Softwareentwicklung gegeben.[381] Diese Dynamik kann die Wichtigkeit einer Anforderung, den Zeitpunkt des Auftretens einer neuen Anforderung und die Häufigkeit von Anforderungsänderungen betreffen. Im Bereich der Softwareentwicklung werden des Weiteren laufend neue Technologien entwickelt,[382] die zusätzlichen Wandlungsdruck erzeugen bzw. neue Umsetzungsalternativen eröffnen, die wiederum Wandlungsdruck hervorru-

---

373 Vgl. Gronau /Wandlungsfähige Informationssystemarchitekturen/ S. 39; Kirchmer, Scheer /Change Management/ S. 3-5. Gronau unterscheidet vielfältige Auslöser zur Reorganisation und teilt diese in Kategorien ein, z. B. Kategorien mit Bezug zur Gebildestruktur, zur Prozessstruktur und zur Umwelt eines Unternehmens.

374 Vgl. Cummings, Worley /Organization/ S. 22; Kirchmer, Scheer /Change Management/ S. 5.

375 Vgl. Kirchmer, Scheer /Change Management/ S. 5.

376 Vgl. Kirchmer /Einführung/ S. 43.

377 Vgl. Sommerville /Sofware Engineering/ S. 589.

378 Vgl. zu Lebenszykluskonzepten beispielsweise Mellis /Projektmanagement/ S. 79-101; Sommerville /Software Engineering/ S. 56-67; Laudon, Laudon /Business/ S. 387-390; Zuser, Grechenig, Köhle /Software Engineering/ S. 70-76.

379 Zu einer Abgrenzung zwischen Individual- und Standardsoftware vgl. Mellis /Projektmanagement/ S. 5-6.

380 Vgl. Oestereich u. a. /Objektorientierung/ S. 22; Naur /Computing/ S. 42

381 Vgl. Mellis u. a. /Software/ S. 11; Brandt-Pook u. a. /Anwendungsentwicklung/ S. 249; Hoch u. a. /Secrets/ S. 99; Weltz, Ortmann /Softwareprojekt/ S. 31.

382 Vgl. Mellis u. a. /Software/ S. 12.

fen können. All diese auftretenden Wirkungen bedingen schon während der Entwicklung eine Wandlungsfähigkeit der zu erstellenden Software.

Die Signifikanz der Wandlungsfähigkeit für Standardsoftware kann im ersten Zeitabschnitt für den Anbieter als auch für den Nachfrager unterschieden werden. Ist ein Anbieter dem Marktdruck durch die Erstellung von Standardsoftware unterworfen, wird eine hohe Wandlungsfähigkeit und infolgedessen durch die Erleichterung des Wandlungsprozesses eine schnelle Reaktionsfähigkeit auf Änderungen von Kundenwünschen wichtiger, als die früh festgelegten Anforderungen umzusetzen.[383] Für Nachfrager kann beobachtet werden, dass Standardsoftware häufig eingesetzt wird, ohne die Möglichkeit zu nutzen, Anpassungen vorzunehmen. Solche Systeme, die oft aus Gründen der kostengünstigen Einführung eingesetzt werden, führen oft dazu, dass die Unternehmensstruktur oder Geschäftsprozesse an die entsprechende Standardsoftware angepasst werden.[384] Geht man davon aus, dass Unternehmen nicht grundsätzlich unwirtschaftlich arbeiten[385] und Strukturen sowie Prozesse derart gestalten, dass ein wettbewerbsfähiger Nutzen entsteht,[386] könnte ein Anpassungszwang des Unternehmens an die Prozesse einer Standardsoftware den wirtschaftlichen Erfolg eines Unternehmens verringern, weil u. U. weniger wirtschaftlich angepasste Strukturen und Prozesse verwendet werden müssen.[387] Folgerichtig ist die Wandlungsfähigkeit einer Standardsoftware sowohl bei Anbietern als auch bei Nachfragern relevant zur Bewertung einer Softwarearchitektur.

Weiterhin wird seit einiger Zeit diskutiert, dass unter bestimmten Bedingungen Flexibilität während der Entwicklung von (insbesondere Standard-)Software Vorteile bringt.[388] Flexible Modelle der Softwareentwicklung wurden entwickelt und eingesetzt, um ent-

---

[383] Vgl. Mellis u. a. /Software/ S. 4-5.

[384] Vgl. Irani, Themistocleous, Love /Impact of EAI/ S. 180; Kirchmer /Einführung/ S. 27 & S. 39.

[385] Dieser Annahme kann gefolgt werden, weil man in einem markwirtschaftlich orientierten Wirtschaftssystem annehmen kann, dass sich das Unternehmen ansonsten nicht am Markt behaupten kann und nicht existieren würde.

[386] Vgl. Kirchmer /Einführung/ S. 1.

[387] Vgl. Kirchmer /Einführung/ S. 15; Irani, Themistocleous, Love /Impact of EAI/ S. 180. Kirchmer spricht hier auch von einer technik-zentrierten gegenüber einer organisations-zentrierten Einführung, vgl. Kirchmer /Einführung/ S. 26.

[388] Vgl. zum folgenden Absatz Mellis /Projektmanagement/ S. 305-324 in Zusammenhang mit 373-374.

sprechende Vorteile realisieren zu können.[389] Es lässt sich jedoch nicht ausschließen, dass auch nach Abschluss der Erstimplementation Änderungen auftreten.[390] Deswegen ist die Fähigkeit, in Wartungsaktivitäten oder späten Entwicklungsstufen einer Software Änderungen umsetzen zu können, von hoher Bedeutung.[391]

Der zweite Zeitabschnitt, der Einsatz und Wartungszeitraum, ist ebenso Wandlungsdruck unterworfen. Weil Unternehmen stetigem Wandel, internen und externen Charakters[392] (vor allem den externen: des Marktes und der Geschäftsumfelder)[393] ausgesetzt sind, müssen sie sich entsprechend anpassen,[394] um die Existenz und den Erfolg des Unternehmens sicherzustellen.[395] Die Unternehmens-IT wiederum hat den Zweck, die Aufgaben des Unternehmens zu unterstützen,[396] somit ist die Unternehmens-IT ebenso Wandel unterworfen und muss sich entsprechend anpassen können.[397] Die Anforderungen bei solchen Anpassungen sind vielfältig: Die Unternehmens-IT muss sich den Änderungen eines Unternehmens, z. B. entsprechend der Struktur als auch der Strategie gerecht werdend, anpassen können.[398] Treffend kann in diesem Zusammenhang Ian Sommerville zitiert werden: „Systeme zu erstellen, die nicht geändert werden müssen, ist (unabhängig von ihrer Größe) unmöglich.“[399] Die Signifikanz der Wandlungsfähigkeit wird erhöht, weil das nachträgliche Einbauen einer Anforderung in eine Software aufwändiger und teurer ist als eine direkte Einführung während der Entwicklungszeit. Die Wartungskosten sind deswegen oft höher als die Kosten der Erstentwicklung.[400]

---

[389] Vgl. Laudon, Laudon /Business/ S. 390-409; Mellis /Turbulent Times/ S. 278-279. Beispielsweise auch Ansätze des Synch&Stabilize und Extreme Programming, zusammenfassend dargestellt in Mellis /Projektmanagement/ S. 374.

[390] Man kann sogar die begründete Behauptung aufstellen, dass Änderungen (zumindest Änderungswünsche) nach Erstimplementation die Regel sind, vgl. Sommerville /Software Engineering/ S. 609-612.

[391] Vgl. Mellis /Projektmanagement/ S. 319-320.

[392] Vgl. Oxman, Smith /Structual Change/ S. 82.

[393] Vgl. Lucas /Information Technology/ S. 127.

[394] Vgl. Kieser, Walgenbach /Organisation/ S. 406-407; Baumöl, Winter /Qualifikation/ S. 46.

[395] Vgl. Pfau /Informationsmanagement/ S. 1.

[396] Vgl. Kapitel 2.2.5.1.

[397] Vgl. Sommerville /Sofware Engineering/ S. 589-591.

[398] Vgl. Oxman, Smith /Structual Change/ S. 77; Akkermans u. a. /Impact/ S. 285-289.

[399] Vgl. Sommerville /Sofware Engineering/ S. 609.

[400] Vgl. Sommerville /Sofware Engineering/ S. 616-617.

Weil Neuentwicklungen bzw. Neueinführungen beim Auftreten neuer Anforderungen im Vergleich zur Wandlung bestehender Systeme nicht vertretbar hohen Aufwand erzeugen würden, muss eine Unternehmens-IT wandlungsfähig sein. Die Unternehmens-IT muss hier insbesondere durch ihren Charakter als integraler Bestandteil eines Unternehmens einen essentiellen Beitrag zur Wandlungsfähigkeit liefern, weil hierdurch vielfältiger Wandlungsbedarf auf die Unternehmens-IT wirkt.[401] Diese Situation wird heute noch verschärft, weil die Häufigkeit von Fällen steigt, in denen schnell wirksame Anpassungsmaßnahmen getroffen werden müssen.[402] Eine Bestätigung für diesen Zustand ist z. B. in der Einigkeit aktueller Literatur ablesbar, wonach der größte Teil der Investitionen in ein Softwaresystem nach dessen Einführung entsteht.[403] Darüber hinaus verstärkt sich diese Problematik, weil sich der Wartungszeitraum einer Unternehmens-IT durchaus auf viele Jahre ausdehnen kann.[404] Der Wandlungsbedarf ist tendenziell höher als bei geringen Systemlebenszeiten. Die Wandlungsfähigkeit wird konsequenterweise auch das ‚Problem bestehender Systeme' genannt.[405]

Die Signifikanz der Wandlungsfähigkeit über den gesamten Lebenszyklus lässt sich zusätzlich durch die Forderung nach einer schnelleren Umsetzungsfähigkeit (neuer) Anforderungen darstellen. In den 90er Jahren hingen Unternehmen mit der Implementation von Anforderungen an ihre Unternehmens-IT zwei bis drei Jahre der Anforderungsformulierung hinterher.[406] Schon damals wurde gefordert, dass die Umsetzung von Anforderungen schneller geschehen müsse und die Softwareentwicklung effizienter zu gestalten sei. Erste Lösungsansätze waren Änderungen in der Vorgehensweise zur Softwareentwicklung und Konzepte wie Extreme Programming[407] oder Rational Unified Process (RUP)[408] wurden entwickelt.[409] Neuere Ansätze verfolgen das Ziel einer schnelleren Softwareentwicklung. Ein wichtiger Ansatz ist dabei die Fähigkeit einer

---

401 Vgl. Lewin, Hunter /Information Technology/ S. 268-271.

402 Vgl. Horváth /Controlling/ S. 4.

403 Vgl. Gronau /Wandlungsfähige Informationssystemarchitekturen/ S. 225 aus dem Jahr 2003 mit Verweisen auf Literatur von 1989 – 1995.

404 Vgl. Sommerville /Software Engineering/ S. 589; Kirchmer /Einführung/ S. 43.

405 Vgl. z. B. Moro, Lehner /Reengineering/ S. 18-19.

406 Vgl. Laudon, Laudon /Business/ S. 389-390; Krcmar /Bedeutung/ S. 401.

407 Vgl. Beck /Extreme Programming/.

408 Vgl. Zuser, Grechenig, Köhle /Software Engineering/ S. 91-96, für eine Übersicht über populäre Vorgehensmodelle vgl. S. 69-104.

Software wandlungsfähig zu sein. Mit diesem Ansatz wird das Ziel verfolgt, weniger Zeit zur Umsetzung neuer Anforderungen zu benötigen.

Signifikanz der Unterqualitätsattribute

- Erweiterbarkeit

  Erweiterbarkeit eines Softwaresystems bedeutet das Einfügen neuer Anforderungen in ein bestehendes Softwaresystem. Es betrifft einerseits bestehende Strukturen einer Software, andererseits ggf. zu erstellende Softwarekomponenten, die mit bestehenden Strukturen zusammenarbeiten können. Bestehende Strukturen werden durch eine Softwarearchitektur beeinflusst – sowohl durch die Gestaltungsvorgabe von Softwarekomponenten als auch von extern sichtbaren Eigenschaften der Softwarekomponenten untereinander und von den Beziehungen der Softwarekomponenten zueinander. Neu zu erstellende Softwarekomponenten sollten sich nach Gestaltungsvorgaben einer Softwarearchitektur richten, um reibungslos mit dem bestehenden Softwaresystem funktionieren zu können.[410] Die Art der Gestaltung durch eine Softwarearchitektur, die den Aufbau eines Softwaresystems festlegt, beeinflusst daher die Erweiterbarkeit einer Software enorm.[411]

- Änderbarkeit

  Änderbarkeit eines Softwaresystems betrifft vorhandene Strukturen einer Software. Die Argumentation der Änderbarkeit verläuft synonym zur Erweiterbarkeit mit dem Unterschied, dass vorhandene Strukturen von Änderungen betroffen sind.

- Testbarkeit

  Der Test eines Softwaresystems bezieht sich mindestens auf den Funktionstest der Softwarekomponenten (Modultest) und den Test des Verbundes der Softwarekomponenten (Integrationstest).[412] Beim Verbundtest sind die Beziehungen der Softwarekomponenten untereinander auch Gegenstand der Tests, weil Softwarekompo-

---

[409] Vgl. Mellis /Turbulent Times/ S. 278-279.

[410] Die Möglichkeit, neu zu erstellende Softwarearchitekturen entgegen der bestehenden Softwarearchitektur zu gestalten, existiert natürlich auch und kann im Einzelfall durchaus sinnvoll sein. Um ein Softwaresystem jedoch verständlich halten zu können und bestehende Richtlinien und Vorgaben umsetzten zu können wird davon ausgegangen, dass bei Erweiterungen i. d. R. die Verwendung existierender Softwarearchitekturvorgaben angestrebt wird.

[411] Vgl. Raasch /Systementwicklung/ S. 31.

[412] Vgl. Sommerville /Software Engineering/ S. 448.

nenten in diesem Test miteinander interagieren müssen. Tests beziehen sich somit mindestens auf Softwarekomponenten und die Beziehungen dieser untereinander. Beides wird durch eine Softwarearchitektur beeinflusst, weshalb die Testbarkeit zur Bewertung einer Softwarearchitektur hinzugezogen wird.

- Stabilität

  Wie beim Unterqualitätsattribut Änderbarkeit dargestellt, beziehen sich Änderungen auf vorhandene Strukturen einer Software. Die Stabilität beschreibt die Fähigkeit einer Software, unbeabsichtigte Auswirkungen durch Änderungen an einer Software zu verhindern. Weil Änderungen die Komponenten einer Software, ihre Beziehungen zueinander oder ihre Präsentation nach außen betreffen, sind diese in starkem Maße von der zugrunde liegenden Softwarearchitektur abhängig. Die Stabilität wird darum durch Softwarearchitekturen beeinflusst und folglich zur Bewertung einer Softwarearchitektur verwendet.

Einstufung der Gewichtung des Qualitätsattributes

Zusammenfassend wird das Qualitätsattribut Wandlungsfähigkeit als wichtig eingestuft und erhält zur Bewertung einer Softwarearchitektur eine hohe Gewichtung.

3.1.3.4 Benutzbarkeit

Das Qualitätsattribut Benutzbarkeit wird von allen vier untersuchten Beiträgen aufgestellt. Die Benutzbarkeit und ihre vorgestellten (Unter-)Qualitätsattribute werden von diesen hauptsächlich aus der Perspektive der Anwender einer Software betrachtet. Der Aufbau einer Software, d. h. insbesondere die zugrunde liegende Softwarearchitektur, ist für den Anwender jedoch intransparent (und i. d. R auch irrelevant). Insofern ist die Bewertung der Benutzbarkeit aus Anwendersicht unabhängig von der verwendeten Softwarearchitektur und kann zur Bewertung einer solchen nur bedingt herangezogen werden. Die Benutzbarkeit wird in dieser Arbeit folglich aus der Perspektive der Software-Entwickler betrachtet.

3.1.3.4.1 Erläuterung der Benutzbarkeit

ISO

Die ISO versteht unter der Benutzbarkeit den Schwierigkeitsgrad, wie die Software verstanden und benutzt und wie deren Anwendung erlernt werden kann, und ob die Software dem Anwender geeignet erscheint, angewendet zu werden.[413] Unterqualitätsattribute der Benutzbarkeit sind:[414] Verständlichkeit (Understandability), Lernfähigkeit (Learnability), Betriebsfähigkeit (Operability), Attraktivität (Attractiveness[415]) und Gehorsamkeit (Compliance).

Boehm

Boehm definiert Benutzbarkeit als das Ausmaß, zu welchem die Software bei der Nutzung dieser praktikabel und bequem gestaltet ist.[416] Boehm unterscheidet zwei Aspekte der Benutzbarkeit: Benutzbarkeit aus Nutzersicht und Benutzbarkeit aus Entwicklersicht.

Raasch

Benutzbarkeit nach Raasch umfasst alle Eigenschaften, die ein einfaches und angenehmes sowie effektives und fehlerarmes Arbeiten mit einer Software gestatten.[417]

Boehm und *Raasch* identifizieren beide das Unterqualitätsattribut Verständlichkeit. Dieses wird bei den Autoren der Wartbarkeit untergeordnet. Weil in dieser Arbeit Benutzbarkeit aus der Perspektive der Software-Entwickler untersucht wird, wird dieses Unterqualitätsattribut an dieser Stelle untersucht.[418]

---

[413] Vgl. ISO /9126/ S. 9.

[414] Vgl. ISO /9126/ S. 7.

[415] Unter der ‚Attraktivität' versteht die ISO: „The capability of the software product to be attractive to the user." (ISO /9126/ S. 10). Diese Definition ist sehr unspezifisch und allgemein gehalten. Die ‚Attraktivität' wird in dieser Arbeit als Unterqualitätsattribut der Gebrauchsgerechigkeit angesehen. Da die Gebrauchsgerechtigkeit zur Bewertung einer Softwarearchitektur irrelevant ist und in dieser Arbeit nicht verwendet wird (vgl. Kapitel 3.1.3.1.1), kann an dieser Stelle auf eine Klärung verzichtet werden.

[416] Vgl. zum folgenden Absatz Boehm u. a. /Characteristics/ S. (3-12)-(3-13).

[417] Raasch /Systementwicklung/ S. 24-25.

[418] Vgl. zur Argumentation Kapitel 3.1.3.3.1.

McCall

McCall versteht unter der Benutzbarkeit den Aufwand, der betrieben werden muss, um die Anwendung einer Software zu erlernen, um eine Software zu bedienen, Eingaben in eine Software zur Eingabe vorzubereiten und Ausgaben einer Software zu interpretieren.[419] McCall identifiziert die Unterqualitätsattribute Betriebsfähigkeit (Operability), Training (Training), Kommunikationsfähigkeit (Communicativeness), Ein-/Ausgabevolumen (Input / output volume) und Ein-/Ausgaberate (Input / output rate)

In Tab. 3-5 sind alle Unterqualitätsattribute der untersuchten Beiträge zur Benutzbarkeit aufgelistet.

ISO 9126	*Boehm*	*Raasch*	*McCall*
Verständlichkeit, Lernfähigkeit, Betriebsfähigkeit, Attraktivität, Gehorsamkeit	Benutzbarkeit aus Nutzersicht, Benutzbarkeit aus Entwicklersicht, *Verständlichkeit*	*Verständlichkeit* [ansonsten keine explizite Nennung von Unterqualitätsattributen]	Betriebsfähigkeit, Training, Kommunikationsfähigkeit, Ein-/Ausgabevolumen, Ein-/Ausgaberate

Tab. 3-5: Unterqualitätsattribute der Benutzbarkeit[420]

Die folgenden Qualitätsattribute lassen sich der Benutzbarkeit aus Anwendersicht zuordnen und werden deswegen im Qualitätsmodell dieser Arbeit nicht betrachtet:

- Lernfähigkeit/Training
  Lernfähigkeit ist die Fähigkeit einer Software, dem Benutzer beibringen zu können, wie die Software zu benutzen ist.[421] Diese Fähigkeit ist von der konkreten Umsetzung funktionaler Anforderungen, der Benutzerführung oder der Programmdoku-

[419] Vgl. McCall /Quality factors/ S. 961.

[420] Zur Verständlichkeit bei Boehm und Raasch siehe Text.

[421] Vgl. ISO /9126/ S. 9; McCall /Quality factors/ S. 965.

mentation[422] abhängig. Diese werden jedoch nicht durch eine Softwarearchitektur beeinflusst[423] und sind zur Bewertung einer Softwarearchitektur nicht geeignet.

- Attraktivität

  Attraktivität ist die Eigenschaft einer Software, für den Benutzer attraktiv zu sein.[424] Hierbei werden z. B. Aspekte des Grafikdesigns der Benutzerschnittstelle bewertet. Die Attraktivität einer Software wird durch Eigenschaften der Bedienoberfläche beeinflusst. Mittlerweile kann man davon ausgehen, dass Softwarearchitekturen die Gestaltungsmöglichkeiten einer Bedienoberfläche nicht mehr beeinflussen, weil die Bedienoberfläche heute auf unterschiedliche technische Art, z. B. durch Nutzung von Metasprachen wie HTML oder durch moderne Gestaltungshilfen wie Java-Swing oder vergleichbaren Technologien – als Basis einer Bedienoberfläche für jede Softwarearchitektur ansprechend – verwendet werden kann. Deshalb wird die Attraktivität nicht zur Bewertung einer Softwarearchitektur verwendet.

- Benutzbarkeit aus Nutzersicht

  Benutzbarkeit aus Nutzersicht nach Boehm stellt die Fähigkeit der Schnittstellen einer Software dar, durch den Menschen komfortabel genutzt und für diesen praktikabel zu sein.[425] Das Unterqualitätsattribut Attraktivität bezieht sich auf das Aussehen einer Bedienoberfläche, wohingegen die Benutzbarkeit aus Nutzersicht bewertet, inwieweit die Bedienoberfläche die Nutzung der Funktionalität in der vorgesehenen Art und Weise ermöglicht. Auch hier kann heute davon ausgegangen werden, dass alle Anforderungen an die Benutzbarkeit aus Nutzersicht durch die vorhandenen Technologien erfüllt werden können und unabhängig von der zugrunde liegenden Softwarearchitektur (mindestens eine Technologie) eingesetzt und die Benutzbarkeit somit gewährleistet werden kann. Daher wird die Benutzbarkeit aus Nutzersicht nicht zur Bewertung einer Softwarearchitektur verwendet.

- Benutzbarkeit aus Entwicklersicht

  Benutzbarkeit aus Entwicklersicht nach Boehm trifft auf den ersten Blick die Perspektive der Benutzbarkeit nach der Auffassung dieser Arbeit. Genau genommen ist

---

[422] Balzert /Entwicklung/ S. 49; Rupietta /Benutzerdokumentation/S. 15-16; Dumke /Software Engineering/ S. 102.

[423] Zur Argumentation funktionaler Anforderungen vgl. Kapitel 3.1.3.2.

[424] Vgl. ISO /9126/ S. 10.

Benutzbarkeit aus Entwicklersicht nach Boehm die Ausprägung einer Software, so vorbereitet und gestaltet zu sein, dass sie in neuen Gebrauchsmöglichkeiten eingesetzt werden kann.[426] Dieses Qualitätsattribut wird in dieser Arbeit synonym zum Qualitätsattribut ‚Wiederverwendbarkeit' (welches Boehm nicht aufstellt) aufgefasst und bewertet.[427]

- Gehorsamkeit

  Gehorsamkeit ist die Fähigkeit einer Software, Standards, Vereinbarungen, Designvorgaben oder Regeln einhalten zu können, die Bezug zur Benutzbarkeit (aus Entwicklersicht) haben.[428] Hierunter fallen z. B. Codierungsrichtlinien oder Namenskonventionen. Eine Softwarearchitektur hat auf solche Standards oder Vereinbarungen keinen Einfluss, weil diese Details der Programmierung betreffen, wohingegen eine Softwarearchitektur die Ebene der Softwarekomponenten betrachtet. Eine Softwarearchitektur kann des Weiteren selber Designvorgaben bzgl. Softwarekomponenten und der Beziehungen dieser zueinander aufstellen. Eine Bewertung einer Softwarearchitektur gegen durch sie selber aufgestellte Designregeln würde selbstverständlich eine positive Bewertung erfahren. Aus diesen Gründen ist die Gehorsamkeit zur Bewertung einer Softwarearchitektur nicht relevant.

Aus diesen Überlegungen leitet sich die Definition des folgenden Kapitels ab.

3.1.3.4.2 Arbeitsdefinition Benutzbarkeit

Benutzbarkeit ist die Fähigkeit einer Software, den Software-Entwickler darin zu unterstützen, Teile oder eine gesamte Software zur Entwicklung verwenden zu können.[429]

---

[425] Vgl. zum folgenden Absatz Boehm u. a. /Characteristics/ S. (3-12)-(3-13).

[426] ebd.

[427] Vgl. zum Qualitätsattribut ‚Wiederverwendbarkeit' Kapitel 3.1.3.7. In dieser Arbeit wird zwar das Qualitätsattribut ‚Benutzbarkeit' auf die Benutzbarkeit aus Entwicklersicht fokussiert, jedoch geht das hier vertretene Verständnis der Benutzbarkeit aus Entwicklersicht über das der Boehmschen ‚Benutzbarkeit aus Entwicklersicht' hinaus, weil nicht nur ein Wiederverwendungscharakter untersucht wird, sondern auch gefragt wird, inwiefern eine Softwarearchitektur Software-Entwicklern Unterstützung bei der Implementierung bieten kann.

[428] Vgl. ISO /9126/ S. 10.

[429] Boehm kommt dieser Definition mit der Auffassung der Benutzbarkeit aus Entwicklersicht am nächsten. Vgl. Boehm u. a. /Characteristics/ S. (3-12)-(3-13).

Unterqualitätsattribute der Benutzbarkeit, die durch Softwarearchitekturen beeinflusst werden können,[430] sind:

- Verständlichkeit

  Verständlichkeit stellt die Fähigkeit einer Software dar, dem Software-Entwickler eindeutig verständlich zu machen, was die Software leisten kann und wie diese für eine zu erfüllende Aufgabe eingesetzt werden kann.[431]. Je kürzer die Zeit, die benötigt wird, um eine Software zu verstehen, desto verständlicher ist diese.[432] Die Unterqualitätsattribute nach McCall[433] Selbstbeschreibung, Einfachheit, Prägnanz, Konsistenz und Modularität sowie ‚Analysierbarkeit' aus dem ISO-Standard 9126 haben Einfluss auf die Verständlichkeit einer Software, weil jeweils hohe Ausprägungen zu einer schnelleren Auffassung beitragen. Diese Unterqualitätsattribute werden in dieser Arbeit nicht mehr explizit als Unter-Unterqualitätsattribute aufgestellt, weil dies eine zu feine Detaillierung darstellen würde.

- Betriebsfähigkeit

  Betriebsfähigkeit ist die Fähigkeit einer Software, den Software-Entwickler in der Anwendung und Kontrolle der Software zu unterstützen.[434]

- Kommunikationsfähigkeit

  Kommunikationsfähigkeit bedeutet, dass die Software nützliche Eingabe- und Ausgabeformate zur Kommunikation verwendet.[435] Hierunter fallen auch die Aspekte des Ein-/Ausgabevolumens und der Ein-/Ausgaberate,[436] die eine Anwendung der Software im intendierten Umfang erlauben sollten.

3.1.3.4.3 Signifikanz der Benutzbarkeit

Die Benutzbarkeit (aus Entwicklersicht) ist für eine Software von hoher Bedeutung. Im Kapitel 3.1.3.3 (Wandlungsfähigkeit) wurde dargestellt, dass Software permanentem

---

[430] Zur Argumentation vgl. Kapitel 3.1.3.4.3.

[431] Vgl. ISO /9126/ S. 9; Boehm u. a. /Characteristics/ S. (3-4)-(3-6); Raasch /Systementwicklung/ S. 31.

[432] Vgl. Raasch /Systementwicklung/ S. 31.

[433] Vgl. zu diesen Tab. 3-4.

[434] Vgl. ISO /9126/ S. 9; McCall /Quality factors/ S. 965.

[435] Vgl. McCall /Quality factors/ S. 965.

[436] Vgl. McCall /Quality factors/ S. 965.

Wandel unterliegt und an geänderte Rahmenbedingungen und Anforderungen angepasst werden muss. Solche Anpassungen finden durch Softwareentwicklung statt und werden von Software-Entwicklern durchgeführt. Software-Entwickler müssen deswegen permanent in der Lage sein, die zu ändernde Software zur Anpassung verwenden zu können. Darunter fällt sowohl die Verwendung eines Teils der Software zur Änderung als auch die Verwendung eines Teils der Software zur Änderung eines anderen Teils, z. B. um den einen Teil um den anderen zu erweitern.

Signifikanz der Unterqualitätsattribute

- Verständlichkeit als Unterqualitätsattribut ist zur Bewertung einer Softwarearchitektur wichtig, weil Softwaresysteme und die Softwareentwicklungsaufgabe in Unternehmen immer umfangreicher und komplexer werden.[437] Damit Software-Entwickler den Überblick über die Struktur der zu erstellenden Software, die durch eine Softwarearchitektur vorgegeben wird, behalten und damit sich Software-Entwickler in einzelnen Teilen einer Software zurechtfinden können, ist eine hohe Verständlichkeit der Software notwendig. Verständlichkeit betrifft die Verständlichkeit (1) der Softwarekomponenten, (2) der Eigenschaften der Softwarekomponenten und (3) der Beziehungen der Softwarekomponenten untereinander. Alle Gegenstände der Verständlichkeit werden durch eine Softwarearchitektur beeinflusst. Aus diesen Gründen wird die Verständlichkeit zur Bewertung einer Softwarearchitektur verwendet.
- Betriebsfähigkeit (von zumindest Teilen) einer Software ist wichtig, damit Software-Entwickler eine Software weiterentwickeln, oder Teile einer Software und die gesamte Software in anderem Kontext verwenden können. Die Betriebsfähigkeit einer Software in Teilen betrifft die Softwarekomponenten, deren Gestaltung durch eine Softwarearchitektur festgelegt wird.
- Kommunikationsfähigkeit einer Software ist wichtig, damit Software-Entwickler Informationen herausfinden können, die sie zur Weiterentwicklung und Wiederverwendung einer Software (bzw. eines Teils einer Software) benötigen. Diese Informationen sind, welche Software (bzw. welcher Teil einer Software) welche Funktion unter welchen Voraussetzungen durchführt. Solche Informationen werden bezüglich einzelner Softwarekomponenten benötigt und betreffen die Eigenschaften und Be-

ziehungen der Softwarekomponenten. Diese werden durch eine Softwarearchitektur festgelegt, und deshalb wird die Kommunikationsfähigkeit zur Bewertung einer Softwarearchitektur verwendet.

Einstufung der Gewichtung des Qualitätsattributes

Zusammenfassend fließt die Bewertung der Benutzbarkeit mit hoher Gewichtung in die Bewertung einer Softwarearchitektur ein.

3.1.3.5 Verlässlichkeit

Das Qualitätsattribut Verlässlichkeit wird von allen vier Beiträgen als Qualitätsattribut aufgestellt. Darüber hinaus wird die Verlässlichkeit von Software von anderen Autoren, z. B. Laprie[438] und Sommerville[439], unabhängig von einem Qualitätsmodell untersucht.

3.1.3.5.1 Erläuterung der Verlässlichkeit

ISO

Nach der ISO stellt die Verlässlichkeit die Fähigkeit der Software dar, eine spezifizierte zu erbringende Dienstleistung unter spezifizierten Bedingungen einhalten zu können.[440] Unterqualitätsattribute der Verlässlichkeit sind Ausgereiftheit (Maturity), Fehlertoleranz (Fault tolerance), Wiederherstellbarkeit (Recoverability) und Gehorsamkeit (Compliance).[441]

Boehm

Verlässlichkeit nach Boehm ist die Fähigkeit einer Software, Funktionen in der erwarteten Art und Weise durchführen zu können.[442] Zwei Fähigkeiten tragen zur Verlässlichkeit bei: Erstens die Fähigkeit, die Anforderungen zu erfüllen, und zweitens die Fähig-

---

[437] Vgl. Mellis /Projektmanagement/ S. 45-46; Posch, Birken, Gerdom /Basiswissen/ S. 4-5.

[438] Vgl. Laprie /Dependability/ S. 1-44.

[439] Vgl. Sommerville /Sofware Engineering/ S. 365-366.

[440] Vgl. ISO /9126/ S. 8.

[441] Vgl. ISO /9126/ S. 7.

[442] Vgl. zum folgenden Absatz Boehm u. a. /Characteristics/ S. (3-14)-(3-15).

keit, das erwartete Verhalten auch bei unerwarteten und nicht getesteten Eingaben auszuführen.

Raasch

Verlässlichkeit ist gegeben, wenn das System die geforderten Leistungen erbringt, ohne fehlerhaft in gefährliche oder sonst unerwünschte Zustände zu gelangen.[443] Ein Unterqualitätsattribut nach Raasch ist Robustheit.

McCall

McCall versteht unter der Verlässlichkeit die Ausprägung, bis zu welchem Maß eine Software ihre vorgesehene Funktion in der jeweils geforderten Genauigkeit durchführen kann.[444] Unterqualitätsattribute nach McCall sind:[445] Fehlertoleranz (Error tolerance), Konsistenz (Consistency), Präzision (Accuracy) und Einfachheit (Simplicity).

Tab. 3-6 fasst alle Unterqualitätsattribute der Verlässlichkeit, die von den untersuchten Beiträgen aufgestellt wurden, zusammen.

ISO 9126	*Boehm*	*Raasch*	*McCall*
Fehlertoleranz, Ausgereiftheit, Wiederherstellbarkeit, Gehorsamkeit	Robustheit, Anforderungs- erfüllung	Robustheit	Fehlertoleranz Konsistenz Präzision Einfachheit

Tab. 3-6: Unterqualitätsattribute der Verlässlichkeit

Weil die folgenden Unterqualitätsattribute nicht durch Softwarearchitekturen beeinflusst werden, werden sie in dieser Arbeit nicht untersucht:

- Präzision

  Präzision ist die Eigenschaft einer Software, die festlegt, dass alle Berechnungen und

[443] Vgl. zum folgenden Absatz Raasch /Systementwicklung/ S. 24.

[444] Vgl. McCall /Quality factors/ S. 961.

[445] Vgl. McCall /Quality factors/ S. 963.

Ausgaben einer Software mindestens der geforderten Genauigkeit entsprechen.[446] Diese Eigenschaft kann unabhängig von der verwendeten Softwarearchitektur durch entsprechend sorgfältige Programmierung erreicht werden.

- Konsistenz

  Konsistenz ist die Eigenschaft einer Software, die einheitliches Design (im Sinne von Präsentation und Layout), Implementationsarten und zu verwendende Notationsarten festlegt.[447] Diese Eigenschaften einer Software können unabhängig von der verwendeten Softwarearchitektur festgelegt werden und werden nicht in das Qualitätsmodell dieser Arbeit aufgenommen.

Folgende Unterqualitätsattribute werden in dieser Arbeit in anderem Zusammenhang beurteilt und fließen nicht als Unterqualitätsattribute der Verlässlichkeit ein:

- Anforderungserfüllung

  Die Anforderungserfüllung nach Boehm ist ein Konzept der Verlässlichkeit und bedeutet die Fähigkeit einer Software, die gestellten Anforderungen zu erfüllen.[448] Eine Software muss gestellte Anforderungen erfüllen, um als verlässlich zu gelten; allerdings wird in dieser Arbeit die Anforderungserfüllung schon durch die Funktionserfüllung[449] abgedeckt.

- Einfachheit

  Die Einfachheit nach McCall stellt die Eigenschaft einer Software dar, implementierte Funktionen so verständlich wie möglich anzubieten.[450] Einfachheit in diesem Sinne und in dem für diese Arbeit benötigten Verständnis (nämlich als Einfachheit für den Entwickler) wird in Kapitel 3.1.3.4 (Benutzbarkeit) behandelt.

Aus diesen Überlegungen leitet sich die Definition des folgenden Kapitels ab.

---

[446] Vgl. McCall /Quality factors/ S. 965.

[447] Vgl. McCall /Quality factors/ S. 965.

[448] Vgl. Boehm u. a. /Characteristics/ S. (3-14)-(3-15);

[449] Vgl. Kapitel 3.1.3.2 Funktionserfüllung, S. 66.

[450] Vgl. McCall /Quality factors/ S. 965.

3.1.3.5.2 Arbeitsdefinition der Verlässlichkeit

Verlässlichkeit ist die Fähigkeit einer Software, die es erlaubt, volles Vertrauen in ihre Dienstleistungen zu haben, weil spezifizierte Ausprägungen der zu erbringenden Dienstleistung unter spezifizierten Bedingungen eingehalten werden.[451]

Unterqualitätsattribute der Verlässlichkeit, die durch Softwarearchitekturen beeinflusst werden können,[452] sind:

- Robustheit (Fehlertoleranz, Ausgereiftheit)
  Robustheit stellt die Fähigkeit einer Software dar, auch dann noch korrekt zu funktionieren, wenn Eingabeparameter nicht den durchgeführten Testeingaben entsprechen,[453] wenn Fehlbedienungen einer Software erkannt und z. B. durch entsprechende Meldungen an den Nutzer bearbeitet werden[454] und die Software nach dem Auftreten dieser Fälle weiterhin genutzt werden kann.[455] Die Robustheit umfasst auch, dass sich ein System gegen externe Angriffe, die durch Zufall oder mit Absicht durchgeführt werden, schützen kann[456] und weiterhin funktionsfähig ist, selbst nachdem Fehlerzustände der Software aufgetreten sind,[457] unabhängig davon, ob das System unter normalen oder nicht normalen Umständen ausgeführt wird.[458] Die Robustheit umfasst nach dieser Auffassung auch die Ausgereiftheit nach dem ISO-Standard 9126.[459]
- Wiederherstellbarkeit
  Wiederherstellbarkeit ist die Fähigkeit einer Software, einen spezifizierten Dienst-

---

[451] Vgl. ISO /9126/ S. 8; IEEE /Glossary/ S. 64.

[452] Zur Argumentation vgl. Kapitel 3.1.3.5.3.

[453] Vgl. Boehm u. a. /Characteristics/ S. (3-14)-(3-15); Sommerville /Sofware Engineering/ S. 376-378.

[454] Vgl. Raasch /Systementwicklung/ S. 24.

[455] Vgl. ISO /9126/ S. 9; McCall /Quality factors/ S. 965.

[456] Vgl. Sommerville /Sofware Engineering/ S. 376-378.

[457] Vgl. ISO /9126/ S. 8.

[458] Vgl. Sommerville /Sofware Engineering/ S. 373-376.

[459] Ausgereiftheit nach der ISO bedeutet das Vermeiden von fehlerhaften Zuständen als Ergebnis von Fehlern in der Software. Vgl. ISO /9126/ S. 8. Die dargestellte Auffassung vertritt auch das IEEE, indem unter der Robustheit das korrekte Funktionieren einer Software in Abhängigkeit von Eingabe- und Umweltalternativen definiert wird. Vgl. IEEE /Glossary/ S. 66.

leistungszustand (z. B. ein Performanzlevel oder eine Datenintegrität) zu erreichen, nachdem ein fehlerhafter Zustand eingetreten ist.[460]

- Gehorsamkeit

  Gehorsamkeit in Bezug auf die Verlässlichkeit bedeutet die Fähigkeit einer Software, entsprechende Standards, Vereinbarungen oder Regelungen (z. B. Gesetze) mit Bezug zur Verlässlichkeit einhalten zu können.[461] Beispielsweise werden von Herstellern sicherheitskritischer Systeme wie der Betriebssoftware von Atomkraftwerken oder Flugzeugen Vorgaben gestellt, welche auch die Verlässlichkeit einer Software betreffen.

Ergänzt werden diese Unterqualitätsattribute durch die Verfügbarkeit nach Sommerville und dem IEEE:

- Verfügbarkeit

  Verfügbarkeit ist die Fähigkeit einer Software, Dienste dann an den Nutzer zu liefern, wenn diese nachgefragt werden.[462]

3.1.3.5.3 Signifikanz der Verlässlichkeit

Die Verlässlichkeit ist zur Architekturbewertung ein wichtiges Qualitätsattribut. Unzuverlässige Systeme werden oft nicht verwendet. Bei einem Systemausfall können entstehende Kosten hoch sein und es kann Informationsverlust entstehen.[463] Weiterhin können Softwarearchitekturen hohen Einfluss auf die Verlässlichkeit einer Software haben.[464]

Signifikanz der Unterqualitätsattribute

- Robustheit einer Software ist von hoher Bedeutung für die Bewertung von Softwarearchitekturen, weil die Robustheit einer Software durch eine Softwarearchitektur stark beeinflusst wird. Eine Softwarearchitektur bestimmt, wie einzelne Komponenten einer Software miteinander interagieren. Diese Festlegung bestimmt, wie Fehler

---

[460] Vgl. ISO /9126/ S. 9.

[461] Vgl. ISO /9126/ S. 9.

[462] Vgl. Sommerville /Sofware Engineering/ S. 369; IEEE /Glossary/ S. 5.

[463] Vgl. Sommerville /Sofware Engineering/ S. 365-366.

weiterkommuniziert werden können, welchen Zustand einzelne Softwarekomponenten nach einer fehlerhaften Ausführung annehmen und wie die Kommunikation innerhalb einer Software geschützt werden kann.

- Wiederherstellbarkeit einer Software ist abhängig davon, wie Softwarekomponenten gestaltet sind und in welcher Beziehung diese zueinander stehen, weil sowohl Softwarekomponenten als auch die Beziehungen der Softwarekomponenten zueinander nach einem Ausfall wiederhergestellt werden müssen. Eine Softwarearchitektur beeinflusst folglich die Wiederherstellbarkeit einer Software.
- Verfügbarkeit ist ein wichtiges Unterqualitätsattribut der Verlässlichkeit, weil eine Dienstleistung genau dann abrufbar sein muss, wenn diese benötigt wird. Weil eine Softwarearchitektur festlegt, wie die Beziehung von Softwarekomponenten zueinander gestaltet ist, hat sie Einfluss auf die Möglichkeiten, wie eine Software auf Dienstleistungsnachfragen reagieren kann.
- Gehorsamkeit bezüglich Verlässlichkeit ist wichtig zur Bewertung einer Softwarearchitektur, weil die Verlässlichkeit häufig das wichtigste Kriterium für Software darstellt – insbesondere für sicherheitskritische Software trifft dies zu.[465] Vorgaben und Standards müssen durch die Software erfüllt werden können. Diese können sogar gesetzlich vorgegeben bzw. geregelt sein. Verlässlichkeit wird für eine gesamte Unternehmens-IT festgelegt.[466] Komponenten müssen somit so miteinander in Beziehung stehen können, dass Vorgaben und Standards entsprechend erfüllt werden können. Eine Softwarearchitektur hat deswegen Einfluss auf die Gehorsamkeit, weil diese die Beziehung der Softwarekomponenten zueinander regelt.[467]

---

[464] Vgl. Posch, Briken, Gerdom /Basiswissen/ S. 77-78.

[465] Vgl. Sommerville /Software Engineeering/ S. 366.

[466] Ggf. auch für Teile einer Unternehmens-IT, die jedoch über einzelne Services hinausgeht.

[467] Beispielsweise wäre eine ereignisgesteuerte Softwarearchitektur zur Flugzeugsteuerung wenig geeignet: Wenn der Pilot einen Steuerbefehl nach links gibt, die Software jedoch nicht sicherstellt, dass der Befehl auch bei allen mechanischen Steuerungskomponenten ankommt, erfüllt diese entsprechende Anforderungen der Flugsicherheitsbehörden nicht.

Einstufung der Gewichtung des Qualitätsattributes

Insgesamt erhält das Qualitätsattribut Verlässlichkeit in dieser Arbeit ein hohes Gewicht. Diese Einschätzung wird durch die Praxis unterstützt, in der berichtet wird, dass hohe Verfügbarkeit wichtig ist und sogar ein K.o.-Kriterium sein kann.[468]

3.1.3.6 Effizienz

Das Qualitätsattribut Effizienz wird von allen vier betrachteten Beiträgen in den jeweiligen Qualitätsmodellen aufgeführt.

3.1.3.6.1 Erläuterung der Effizienz

ISO

Die ISO versteht unter der Effizienz die Fähigkeit einer Software, eine angemessene Performanz in Relation zu eingesetzten Ressourcen unter festgelegten Bedingungen bereitzustellen.[469] Unterqualitätsattribute sind Antwortzeitverhalten (Time behaviour), der Grad der Ressourcenausnutzung (Resource utilisation) und Gehorsamkeit (Compliance).[470]

Boehm

Effizienz ist gegeben, wenn eine Software ihre vorgesehenen Funktionen ohne Ressourcenverschwendung durchführt.[471] Boehm bezieht sich in seinen Ausführungen stark auf Hardware. Problematisch bei dieser Definition ist der Begriff der Ressourcenverschwendung.

---

[468] Vgl. Kapitel 6.2.1; Moro, Lehner /Reengineering/ S. 35;

[469] Vgl. ISO /9126/ S. 10. Die ISO verwendet den Ausdruck ‚angemessen' („The capability of the software product to provide appropriate performance [...]") und bleibt dadurch ungenau.

[470] Vgl. ISO /9126/ S. 7.

[471] Boehm u. a. /Characteristics/ S. (3-15)-(3-16).

Raasch

„Effizienz ist das Ausmaß der Inanspruchnahme von Betriebsmitteln (Hardware) durch das Software-Produkt bei gegebenem Funktionsumfang.“[472] Raasch geht auch auf den Zeitbedarf ein (neben Speicher, Durchsatz und Antwortzeitverhalten → Hardware- vs. Software-Effizienz).

McCall

McCall beurteilt bei der Bewertung der Effizienz die Menge von Computerressourcen, die von einer Software benötigt werden, um eine Funktion auszuführen.[473] Als Kriterien zieht er die benötigte Rechenleistung und den Datenbedarf heran, weshalb die Unterqualitätsattribute nach McCall die Laufzeiteffizienz (Execution efficiency) und Speichereffizienz (Storage efficiency) sind.[474]

In Tabelle Tab. 3-7 sind alle Unterqualitätsattribute der untersuchten Beiträge zur Effizienz aufgelistet.

ISO 9126	*Boehm*	*Raasch*	*McCall*
Antwortzeitverhalten, Grad der Ressourcenausnutzung, Gehorsamkeit	[keine explizite Nennung von Unterqualitätsattributen]	Hardware-, Softwareeffizienz	Laufzeiteffizienz Speichereffizienz

Tab. 3-7: Unterqualitätsattribute der Effizienz

Aus diesen Überlegungen leitet sich die Definition des folgenden Kapitels ab.

[472] Vgl. Raasch /Systementwicklung/ S. 23-24.

[473] Vgl. McCall /Quality factors/ S. 961.

[474] Vgl. McCall /Quality factors/ S. 963.

3.1.3.6.2 Arbeitsdefinition der Effizienz

Effizienz ist die Fähigkeit einer Software, ihre festgelegte Funktionalität mit einem Minimum an einzusetzenden Ressourcen unter festgelegten Bedingungen bereitzustellen.[475] Unterqualitätsattribute der Effizienz, die durch Softwarearchitekturen beeinflusst werden können, sind:[476]

- Antwortzeitverhalten

  Das Antwortzeitverhalten gibt die Fähigkeit der Software an, möglichst geringe Antwort- und Verarbeitungszeiten sowie gleichzeitig hohe Durchsatzraten zu erzielen, wenn Funktionen unter festgelegten Bedingungen ausgeführt werden.[477] Die Laufzeiteffizienz nach McCall[478] sowie die Softwareeffizienz nach Raasch[479] sind synonym zum Antwortzeitverhalten definiert.

- Grad der Ressourcenausnutzung

  Ressourcenausnutzung ist die Fähigkeit einer Software, eine möglichst geringe Menge von Ressourcen zu benutzen, wenn Funktionen unter festgelegten Bedingungen ausgeführt werden.[480] Die Speichereffizienz nach McCall[481] sowie die Hardwareeffizienz nach Raasch[482] sind synonym definiert.

- Gehorsamkeit

  Gehorsamkeit in Bezug auf Effizienz ist die Fähigkeit einer Software, Standards oder Vereinbarungen mit Bezug zur Effizienz einhalten zu können.[483]

---

[475] Vgl. ISO /9126/ S. 10; IEEE /Glossary/ S. 28. Die ISO verwendet den Ausdruck ‚angemessen' („The capability of the software product to provide *appropriate* performance [...]") und bleibt dadurch ungenau. Die Arbeitsdefinition richtet sich nach den weiter oben vorgestellten Autoren und dem IEEE.

[476] Zur Argumentation vgl. Kapitel 3.1.3.6.3.

[477] Vgl. ISO /9126/ S. 10. Welche tatsächlichen Werte als hoch und niedrig angesehen werden, muss fallabhängig im konkreten Praxisprojekt beurteilt werden.

[478] McCall versteht darunter, dass eine minimale Bearbeitungszeit für Funktionen vorliegen soll. Vgl. McCall /Quality factors/ S. 965.

[479] Vgl. Raasch /Systementwicklung/ S. 23-24.

[480] Vgl. ISO /9126/ S. 10.

[481] Vgl. McCall /Quality factors/ S. 965.

[482] Vgl. Raasch /Systementwicklung/ S. 24.

[483] Vgl. ISO /9126/ S. 10.

3.1.3.6.3 Signifikanz der Effizienz

Effizienz wird durch Softwarearchitekturen beeinflusst, weil eine Softwarearchitektur festlegt, in welcher Beziehung Softwarekomponenten zueinander stehen und wie sie gestaltet werden. Dies hat Einfluss auf die Wahl der Einsatzmöglichkeit von Softwarekomponenten und Hardware, wodurch die Möglichkeit der Gestaltung von Softwarekomponenten und Hardware zur Effizienzsteigerung beeinflusst werden können.

Signifikanz der Unterqualitätsattribute

- Antwortzeitverhalten

  Das Antwortzeitverhalten einer Software wird beeinflusst durch Faktoren wie z. B. der Ressourcenausstattung und der Zurverfügungstellung von Ressourcen, durch Optimierung der Implementierung im Hinblick auf Effizienz, durch die Höhe des notwendigen internen Kommunikationsbedarfs und durch die Menge zu verarbeitender Daten. Softwarearchitekturen beeinflussen erstens die Möglichkeiten der Zurverfügungstellung von Ressourcen, weil Softwarekomponenten unterschiedlich gestaltet werden können. Z. B. können Verbünde von Funktionen unterschiedlich zusammengestellt werden. Diese Verbünde können z. T. nicht getrennt werden und müssen deswegen auf gleichen Ressourcen ablaufen. Zweitens beeinflussen Softwarearchitekturen die Höhe interner Kommunikation, zum einen, weil die Beziehungen der Softwarekomponenten untereinander geregelt werden und demgemäß die Art jedes auftretenden Kommunikationsfalls festgelegt wird; zum anderen, weil durch die Festlegung von Gestaltungsrichtlinien für Softwarekomponenten der Bedarf an Kommunikation unter Softwarekomponenten und der Bedarf an Datenaustausch unterschiedlich ausfallen kann. Das Antwortzeitverhalten wird somit durch eine Softwarearchitektur beeinflusst und fließt in die Bewertung einer Softwarearchitektur ein.

- Grad der Ressourcenausnutzung

  Der Grad der Ressourcenausnutzung wird durch eine Softwarearchitektur insofern beeinflusst, weil eine Softwarearchitektur festlegt, wie eine Software in Softwarekomponenten aufgeteilt wird. Der Einfluss entsteht dadurch, dass deswegen festgelegt wird, wie und ob eine Software auf Ressourcen verteilt werden kann.

- Gehorsamkeit

  Gehorsamkeit der Effizienz ist wichtig, weil Softwarearchitekturen auf Effizienz Einfluss haben und die Befolgung von Vereinbarungen im Hinblick auf Effizienz für Software wichtig ist. Wichtig ist sie deswegen, weil sich die Effizienz auf die Benutzbarkeit aus Nutzersicht auswirken kann. Dies zeigt sich daran, dass häufig vertraglich festgelegte Effizienzvereinbarungen getroffen werden, deren Nichteinhaltung zu entsprechend vereinbarten Konsequenzen führen kann.

Einstufung der Gewichtung des Qualitätsattributes

Wie oben dargestellt ist Effizienz zur Bewertung einer Softwarearchitektur relevant. Jedoch ist es möglich, die Effizienz einer Software in Form von Verarbeitungsgeschwindigkeit (unabhängig von der Art der Verarbeitung, also z. B. Durchsatzraten oder Antwortzeitverhalten) auch nachträglich, z. B. durch Hardwareausbau, zu verbessern.[484] Daher fließt Effizienz mit niedrigem Gewicht in die Bewertung einer Softwarearchitektur ein.

3.1.3.7 Wiederverwendbarkeit

Das Qualitätsattribut Wiederverwendbarkeit wird nur von den Autoren Raasch und McCall explizit als Qualitätsattribut aufgestellt.

3.1.3.7.1 Erläuterung der Wiederverwendbarkeit

Raasch

Bereits entwickelte Softwarekomponenten sind wiederverwendbar, wenn sie in anderen Umgebungen, für die sie ursprünglich nicht geplant waren, eingesetzt werden können.[485]

---

[484] Vgl. zum folgenden Absatz Sommerville /Sofware Engineering/ S. 365-366.

[485] Vgl. Raasch /Systementwicklung/ S. 35. Auch das IEEE verwendet eine sehr ähnliche Definition: Wiederverwendbarkeit ist das Ausmaß, zu welchem ein Softwaremodul oder ein anderes Arbeitsprodukt in mehr als einem Softwaresystem verwendet werden kann. Vgl. IEEE /Glossary/ S. 66.

Raasch argumentiert, dass Portabilität Voraussetzung zur Wiederverwendung ist.[486] Er geht davon aus, dass Softwarekomponenten in anderen Softwareumgebungen eingesetzt werden, sobald diese wiederverwendet werden sollen, weil das Merkmal der Portabilität Voraussetzung wird. In dieser Arbeit wird unter Wiederverwendbarkeit jedoch auch die Fähigkeit zur Mehrfachverwendbarkeit einzelner Softwarekomponenten einbezogen. Daher wird es möglich, eine unternehmensinterne Auffassung der Wiederverwendbarkeit zu vertreten. Unternehmensintern ist die Fähigkeit portabel zu sein nicht notwendig, weil die Softwarekomponenten schon lauffähig im Unternehmen eingesetzt werden, sondern die Fähigkeit, die entsprechende Dienstleistung auch anderen Softwarekomponenten anbieten zu können.

McCall

Wiederverwendbarkeit drückt aus, inwiefern eine Software in anderen Anwendungen genutzt werden kann.[487] Unterqualitätsattribute nach McCall sind:[488] Allgemeingültigkeit (Generality), Modularität (Modularity), Softwareunabhängigkeit (Software system independence), Hardwareunabhängigkeit (Machine independence) und Selbstbeschreibung (Self-descriptiveness)

Tab. 3-8 fasst alle Unterqualitätsattribute der Wiederverwendbarkeit, die von den untersuchten Beiträgen aufgestellt wurden, zusammen.

ISO 9126	*Boehm*	*Raasch*	*McCall*
[Nennt das Qualitätsattribut nicht]	[Nennt das Qualitätsattribut nicht]	[keine explizite Nennung von Unterqualitäts-attributen]	Allgemeingültigkeit, Modularität, Softwareunabhängigkeit, Hardwareunabhängigkeit, Selbstbeschreibung

Tab. 3-8: Unterqualitätsattribute der Wiederverwendbarkeit

[486] Vgl. Raasch /Systementwicklung/ S. 36.

[487] Vgl. McCall /Quality factors/ S. 961.

[488] Vgl. McCall /Quality factors/ S. 963.

Weil die folgenden Unterqualitätsattribute nicht durch Softwarearchitekturen beeinflusst werden, werden sie in dieser Arbeit nicht untersucht:

- Softwareunabhängigkeit / Hardwareunabhängigkeit

  Softwareunabhängigkeit stellt das Ausmaß einer Software dar, inwieweit diese von der Softwareumgebung (hierunter fallen z. B. Betriebssysteme oder Middleware), in welcher sie abläuft, unabhängig ist.[489] Hardwareunabhängigkeit stellt das Ausmaß einer Software dar, inwiefern diese von der Hardwareumgebung, in welcher die Software abläuft, unabhängig ist.[490]

  Wie in Kapitel 1.3.3 dargelegt, ist die Betrachtung dieser Arbeit losgelöst von technologischen Aspekten, weil Softwarearchitekturen mittels verschiedener Techniken umgesetzt werden können. Darunter fallen sowohl Softwaretechnologien als auch Hardwaretechnologien. Die Bewertung der Abhängigkeit einer Software von der Software- und Hardwareumgebung hängt direkt von der Wahl der verwendeten Technologien ab. Aspekte der technischen Realisation werden jedoch nicht betrachtet, weil eine Bewertung ansonsten, wie im Fall funktionaler Anforderungen,[491] den Rahmen dieser Arbeit bei weitem übertreffen würde.[492]

Aus diesen Überlegungen leitet sich die Definition des folgenden Kapitels ab.

3.1.3.7.2 Arbeitsdefinition der Wiederverwendbarkeit

Wiederverwendbarkeit ist die Fähigkeit einer Software bzw. Softwarekomponente, in mehr als einem Anwendungsfall Verwendung zu finden.[493]

Diese Arbeitsdefinition wird gewählt und der Definition von Raasch wird in dieser Arbeit nicht gefolgt, weil Raasch eine geplante Wiederverwendung explizit ausschließt. Gleichzeitig schreibt Raasch, dass Wiederverwendung „nicht in wünschenswertem Um-

---

[489] Vgl. McCall /Quality factors/ S. 965.

[490] Vgl. McCall /Quality factors/ S. 965.

[491] Vgl. Kapitel 3.1.3.2.1.

[492] Vgl. Kapitel 1.3.3.

[493] In Ergänzung zu den weiter oben vorgestellten Definitionen stimmt die Definition des IEEE auch mit dieser Arbeitsdefinition überein. Vgl. IEEE /Glossary/ S. 66 („The degree to which a software module or other work product can be used in more than one computer program or software system.")

fang nachträglich erreichbar [ist]“[494]. Nach dieser Auffassung wäre Wiederverwendbarkeit somit nur zufällig erreichbar. In dieser Arbeit jedoch wird davon ausgegangen, dass Wiederverwendung nicht nur zufällig erreicht, sondern dass Software so geplant werden kann, dass Wiederverwendung erreichbar ist. Unter Umständen jedoch ist zum Planungszeitpunkt der Wiederverwendung nicht bekannt, wann und in welchem Umfang Wiederverwendung stattfinden wird und welche Softwarekomponenten wiederverwendet werden können.

Unterqualitätsattribute der Wiederverwendbarkeit, die durch Softwarearchitekturen beeinflusst werden können,[495] sind:

- Allgemeingültigkeit
  Allgemeingültigkeit stellt das Ausmaß von Softwarekomponenten dar, inwieweit der Funktionsumfang der Softwarekomponente wenig Spezialisierung aufweist.[496]
- Modularität
  Modularität stellt das Ausmaß einer Software dar, inwieweit die einzelnen Softwarekomponenten voneinander unabhängig strukturiert sind.[497]
- Selbstbeschreibung
  Selbstbeschreibung stellt die Fähigkeit einer Software dar, Erläuterungen einer Funktion zur Verfügung zu stellen. McCall definiert Selbstbeschreibung im Hinblick auf die Erläuterung der Implementation einer Funktion.[498] Allerdings wird die Erläuterung der Implementation einer Funktion in dieser Arbeit unter dem Unterqualitätsattribut Verständlichkeit der Benutzbarkeit (aus Entwicklersicht) untersucht.[499]

---

[494] Raasch /Systementwicklung/ S. 36.

[495] Zur Argumentation vgl. Kapitel 3.1.3.7.3.

[496] Vgl. McCall /Quality factors/ S. 965.

[497] Vgl. McCall /Quality factors/ S. 965.

[498] Vgl. McCall /Quality factors/ S. 965.

[499] Vgl. hierzu Kapitel 3.1.3.4.

3.1.3.7.3 Signifikanz der Wiederverwendbarkeit

Wiederverwendung sollte in der Softwareentwicklung wie in anderen technisch-konstruktiven Techniken angewendet werden.[500] In diesen Disziplinen basiert der Entwurf auf der Wiederverwendung von Komponenten unterschiedlichen Funktionsumfangs wie z. B. Ventile, Kolben und schließlich Motoren. Die mit der Wiederverwendung in diesen Disziplinen verbundenen Vorteile wie z. B. eine höhere Zuverlässigkeit oder eine beschleunigte Entwicklung können auch auf Softwaresysteme übertragen werden.

Eine Softwarearchitektur legt die Struktur eines Softwaresystems fest.[501] Die Struktur eines Softwaresystems beinhaltet unter anderem die Aufteilung der Software in Softwarekomponenten und wie diese miteinander kommunizieren. Wiederverwendbarkeit innerhalb eines Softwaresystems kann sich auf zwei Bereiche beziehen. Zum einen auf Quellcode, der wiederverwendet wird, in dem dieser an anderer Stelle hineinkopiert wird, zum anderen auf Softwarekomponenten, die in einem weiteren Anwendungsfall noch einmal verwendet werden. Die Wiederverwendbarkeit von Quellcode wird durch eine Softwarearchitektur nicht beeinflusst. Dagegen beeinflusst sie die Möglichkeit der Wiederverwendung auf Softwarekomponentenebene enorm, weil sowohl der Funktionsumfang einer Softwarekomponente als auch die Kommunikation zu anderen Softwarekomponenten durch eine Softwarearchitektur stark beeinflusst werden. Der Funktionsumfang einer Softwarekomponente muss der ‚wieder' nachgefragten Funktion entsprechen, und die Kommunikationsmöglichkeiten müssen für die weitere Verwendung angemessen sein.[502] Um eine Softwarekomponente wiederverwenden zu können, müssen jedoch sowohl der entsprechend gewünschte Funktionsumfang als auch die entsprechend möglichen Kommunikationsfähigkeiten gegeben sein.

Signifikanz der Unterqualitätsattribute

- Allgemeingültigkeit

  Eine Softwarearchitektur hat Einfluss auf die Allgemeingültigkeit, weil der Funkti-

---

[500] Vgl. zum folgenden Absatz Sommerville /Software Engineering/ S. 315-317.

[501] Vgl. Kapitel 2.1.

[502] Sind Kommunikationsfähigkeiten einer Softwarekomponente nicht per se im erforderlichen Umfang vorhanden, kann Wiederverwendung dennoch durch Anpassung möglich sein.

onsumfang, der die Allgemeingültigkeit wiederum bestimmt, durch Softwarearchitekturen beeinflusst wird. Der Einfluss auf die Wiederverwendbarkeit ist, dass allgemeingültige Softwarekomponenten tendenziell häufiger wiederverwendet werden können als sehr stark spezialisierte.

- Modularität

  Durch Vorgaben einer Softwarearchitektur, wie Softwarekomponenten gestaltet werden sollen, wird die Modularität direkt beeinflusst. Ein modularer Aufbau von Softwarekomponenten unterstützt die Wiederverwendbarkeit, weil existierende Teilkomponenten direkt verwendet werden können, ohne dass die Notwendigkeit besteht, den entsprechenden Quellcode so zu kopieren und anzupassen, dass dieser ‚wiederverwendet' werden kann.[503]

- Selbstbeschreibung

  Selbstbeschreibung stellt die Fähigkeit einer Software dar, Erläuterungen einer Funktion zur Verfügung zu stellen. McCall definiert Selbstbeschreibung im Hinblick auf die Erläuterung der Implementation einer Funktion.[504] Allerdings wird die Erläuterung der Implementation einer Funktion in dieser Arbeit unter dem Unterqualitätsattribut Verständlichkeit der Benutzbarkeit (aus Entwicklersicht) untersucht.[505] Eine vollständige und akkurate Selbstbeschreibung einer Software bzw. von Softwarekomponenten ermöglicht eine Wiederverwendung der Software bzw. Softwarekomponente, weil Beschreibungen die Informationen übermitteln, welche Funktion durch eine Software bzw. Softwarekomponente ausgeführt wird. Diese Information ist Grundvoraussetzung zur Wiederverwendung. Weil Softwarearchitekturen die Struktur der Software, insbesondere im Bereich der Softwarekomponenten festlegen, wird die Fähigkeit, vollständige und akkurate Selbstbeschreibungen aufstellen und mitteilen zu können, durch Softwarearchitekturen beeinflusst.

---

[503] Posch, Birken und Gerdom nennen diese Art der Wiederverwendung „Codeplünderung". Die Autoren zeigen, welche Nachteile mit einer „Codeplünderung" verbunden sind: z. B. die Schaffung impliziter Abhängigkeiten und hohe Zeitinvestition durch Such- und Anpassungsaktivitäten. Vgl. Posch, Birken, Gerdom /Basiswissen/ S. 272. Das Kopieren von Quellcode führt ebenso zu Quellcoderedundanzen, die Fehler bedingen können, die Wartung erschweren und die Quellcodestruktur einer Software kompliziert gestalten.

[504] Vgl. McCall /Quality factors/ S. 965.

[505] Vgl. hierzu Kapitel 3.1.3.4.

Wiederverwendbarkeit wird als Möglichkeit angesehen, Entwicklungszeiten zu verringern.[506] Jedoch konnte Software bisher auf breiter Basis noch nicht wiederverwendbar konzipiert werden.[507] Angesichts des Kostendrucks auf Unternehmen, der heute immer mehr steigt,[508] ist Wiederverwendbarkeit eine wichtige Möglichkeit, diese Anforderungen besser erfüllen zu helfen. Mittels Wiederverwendung wird angestrebt Produktions- und Reaktionszeiten zu kürzen, um dadurch Produkte schneller an den Markt bringen und Produktion günstiger gestalten zu können.

Einstufung der Gewichtung des Qualitätsattributes

Weil die Wiederverwendbarkeit durch eine Softwarearchitektur direkt beeinflusst wird und weil ein Potential besteht, das heutigen Anforderungen von Betroffenen der Unternehmens-IT durch Wiederverwendbarkeit besser nachkommen kann, fließt das Qualitätsattribut Wiederverwendbarkeit mit hohem Gewicht in die Bewertung ein.

3.1.3.8 Portabilität

Das Qualitätsattribut Portabilität wird von allen vier untersuchten Beiträgen verwendet.

3.1.3.8.1 Erläuterung der Portabilität

ISO

Portabilität ist die Fähigkeit einer Software, von einer Umgebung in eine andere übertragen werden zu können.[509] Die Umgebung kann dabei Unterschiede in Bereichen der Software, Hardware und Organisation aufweisen. Unterqualitätsattribute der Portabilität dem ISO-Standard 9126 nach sind:[510] Anpassungsfähigkeit (Adaptability), Installationsfähigkeit (Installability), Koexistenz (Co-existence), Ersetzbarkeit (Replaceability) und Gehorsamkeit (Compliance).

---

[506] Vgl. Sommerville /Software Engineering/ S. 317.

[507] Vgl. Raasch /Systementwicklung/ S. 36.

[508] Dies kann seit einigen Jahren sehr gut in der Tagespresse nachverfolgt werden.

[509] Vgl. zum folgenden Absatz ISO /9126/ S. 11.

[510] Vgl. ISO /9126/ S. 7.

Boehm

Portabilität wird aufgefasst als die Fähigkeit, einfach und ordentlich auf unterschiedlichen Computerkonfigurationen funktionsfähig zu sein.[511]

Raasch

Portabilität nach Raasch ist gegeben, wenn ein System mühelos in eine andere Hardware- oder Software-Umgebung eingeführt werden kann.[512]

McCall

Portabilität bedeutet den Aufwand, der betrieben werden muss, um eine Software von einer Hardwarekonfiguration oder von einer Softwaresystemumgebung in eine andere zu übertragen.[513]

In Tab. 3-9 sind alle Unterqualitätsattribute der untersuchten Beiträge zur Portabilität aufgelistet.

ISO 9126	*Boehm*	*Raasch*	*McCall*
Anpassungsfähigkeit, Installationsfähigkeit, Koexistenz, Ersetzbarkeit, Gehorsamkeit	[keine explizite Nennung von Unterqualitätsattributen]	[keine explizite Nennung von Unterqualitätsattributen]	[keine explizite Nennung von Unterqualitätsattributen]

Tab. 3-9: Unterqualitätsattribute der Portabilität

Die folgenden Qualitätsattribute lassen sich der Benutzbarkeit aus Anwendersicht zuordnen und werden deswegen im Qualitätsmodell dieser Arbeit nicht betrachtet:

- Fähigkeit zur Koexistenz
  Koexistenz ist die Fähigkeit einer Software, zusammen mit weiterer voneinander un-

[511] Boehm u. a. /Characteristics/ S. (3-7)-(3-8).

[512] Vgl. Raasch /Systementwicklung/ S. 32.

[513] Vgl. McCall /Quality factors/ S. 961; Das IEEE verwendet eine fast wortgleiche Definition der Portabilität, vgl. IEEE /Glossary/ S. 57.

abhängiger Software in einer gemeinsamen Umgebung und unter Nutzung derselben Ressourcen existieren zu können.[514] Die Fähigkeit zur Koexistenz ist einerseits von der eingesetzten Software- und Hardwareumgebung abhängig, weil diese in der Lage sein muss, mit allen Softwaresystemen, die koexistieren sollen, zurecht zu kommen; andererseits schränkt die getroffene Wahl einer technologischen Umsetzung einer Software ihre Fähigkeit ein, mit einer anderen Software koexistieren zu können, weil sich ggf. verschiedene Anforderungen einer bestimmten Funktionalität an die Softwareumgebung ergeben, die nicht in Übereinstimmung gebracht werden kann. Hierunter fällt z. B., dass Softwarekomponenten, welche auf DCOM aufbauen, nicht unter Unix verwendet werden können. In dieser Arbeit werden jedoch technologische Umsetzungsalternativen nicht beachtet,[515] und folglich wird die Fähigkeit zur Koexistenz nicht zur Bewertung einer Softwarearchitektur herangezogen.[516]

- Gehorsamkeit

  Gehorsamkeit ist die Fähigkeit einer Software, Standards und Vereinbarungen einzuhalten, die im Zusammenhang mit Portabilität existieren.[517] Standards und Vereinbarungen können für ein Unternehmen bezüglich spezifischer Ausprägungen einer entsprechenden Unternehmens-IT-Infrastruktur definiert werden. Z. B. kann festgelegt werden, dass bestimmte Protokolle oder Betriebssysteme verwendbar sein müssen. Allgemeine Standards und Vereinbarungen sind nicht bekannt. Aus diesem Grund ist die Gehorsamkeit für den Rahmen dieser Arbeit nicht relevant. Sie kann jedoch für eine spezifische Bewertung herangezogen werden.

Aus diesen Überlegungen leitet sich die Definition des folgenden Kapitels ab.

---

[514] Vgl. ISO /9126/ S. 11.

[515] Vgl. Kapitel 1.3.3.

[516] Hier sei hervorgehoben, dass eine Bewertung einer Softwarearchitektur in einem konkreten Praxisprojekt, bei dem die technologische Umsetzung festgelegt ist, oder nur zwischen wenigen Umsetzungsalternativen entschieden werden soll, durch die Bewertung der Fähigkeit zur Koexistenz erweitert werden kann.

[517] Vgl. zum folgenden Absatz ISO /9126/ S. 11.

3.1.3.8.2 Arbeitsdefinition der Portabilität

Portabilität ist die Fähigkeit einer Software, von einer Umgebung in eine andere übertragen werden zu können.[518] Dabei kann die Umgebung unterschiedlich sein in Bereichen der Soft- und Hardware sowie der Organisation.

Unterqualitätsattribute, welche durch Softwarearchitekturen beeinflusst werden können,[519] sind:

- Anpassungsfähigkeit
  Anpassungsfähigkeit ist die Fähigkeit einer Software, in unterschiedlichen (spezifizierten) Umgebungen lauffähig zu sein, ohne dass Aktivitäten durchgeführt werden müssen, die nicht explizit zur Anpassung (z. B. eine Neukompilierung für ein zweites Betriebssystem oder der Einsatz sog. virtueller Maschinen[520]) vorgesehen sind.[521]
- Installationsfähigkeit
  Installationsfähigkeit bezeichnet die Fähigkeit einer Software, in unterschiedlichen Umgebungen installiert werden zu können.[522]
- Ersetzbarkeit
  Ersetzbarkeit ist die Fähigkeit einer Software, eine andere Software, die für gleiche Zwecke eingesetzt wurde, in der entsprechenden Umgebung ersetzen zu können.[523]

3.1.3.8.3 Signifikanz der Portabilität

Da Unternehmen heute verstärkt global agieren,[524] und ihre Unternehmensstrukturen entsprechend ausrichten, wird auch die globale Ausrichtung der Unternehmens-IT immer wichtiger. In diesem Zusammenhang lässt sich z. B. Dezentralität als Anforderung

---

[518] Vgl. zum folgenden Absatz ISO /9126/ S. 11.

[519] Zur Argumentation vgl. Kapitel 3.1.3.8.3.

[520] Virtuelle Maschinen stellen eine Laufzeitumgebung zur Verfügung, die eine Software ohne vorzunehmende Anpassungen in verschiedenen Umgebungen (z. B. auf verschiedenen Betriebssystemen oder Hardwareplattformen) funktionsfähig macht. In solchen Fällen muss allerdings die jeweilige virtuelle Maschine den entsprechenden Umgebungen angepasst werden. Ein populäres Beispiel ist Java.

[521] Vgl. ISO /9126/ S. 11.

[522] Vgl. ISO /9126/ S. 11.

[523] Vgl. ISO /9126/ S. 11.

[524] Vgl. Macharzina /Unternehmensführung/ S. 679.

an eine Unternehmens-IT formulieren.[525] Dezentralität wiederum lässt sich durch portable Unternehmens-IT erreichen, weil Softwarefunktion dadurch an unterschiedlichen Orten eingesetzt werden kann.

Portabilität von Software ist vor allem durch den Einsatz mobiler Endgeräte für Unternehmenssoftware sehr wichtig.[526] Durch den Einsatz mobiler Endgeräte wird gegenwärtig versucht, den Mitarbeitern eines Unternehmens Daten und Funktion an dem Ort zur Verfügung zu stellen, an dem diese benötigt werden. Beispielsweise werden Daten bei Paketzustellern heute in nahezu Echtzeit erhoben. Portabilität ist heutzutage vor allem deswegen wichtig, um Verarbeitungspotentiale räumlich verteilt installieren und miteinander verbinden zu können, unter der Anforderung, keine Probleme der Dezentralisierung von Daten und Anwendungen zu erzeugen.[527]

Signifikanz der Unterqualitätsattribute

- Anpassungsfähigkeit

  Die Anpassungsfähigkeit betrifft Softwarekomponenten, weil diese in anderen Umfeldern eingesetzt werden sollen, und Beziehungen zwischen Softwarekomponenten, weil ein Einsatz in einem anderem Umfeld auch Kommunikation mit diesem anderen Umfeld beinhaltet. Softwarearchitekturen beeinflussen diese beiden Aspekte, weshalb Anpassungsfähigkeit zur Bewertung einer Softwarearchitektur verwendet wird.

- Installationsfähigkeit

  Installationsfähigkeit betrifft die Fähigkeit der Softwarekomponenten, in einer anderen Umgebung lauffähig zu sein. Die Gestaltung der Softwarekomponenten ist von einer Softwarearchitektur abhängig, weshalb Installationsfähigkeit zur Bewertung einer Softwarearchitektur verwendet wird.

- Ersetzbarkeit

  Ersetzbarkeit betrifft eine Software (oder Teile dieser, z. B. einzelne Softwarekom-

---

[525] Vgl. Gronau /Wandlungsfähige Informationssystemarchitekturen/ S. 218.

[526] Vgl. Hansen, Neumann /Wirtschaftsinformatik/ S. 539-540; CapGemini /IT-Trends 2005/ S. 26.

[527] Vgl. Frese /Organisation/ S. 131-132. Frese stellt die Probleme zur Anfangszeit der individuellen Datenverarbeitung dar. Auf S. 153 erklärt Frese, dass Entscheidungskompetenzen heute von informationstechnischen Potentialen abhängen. Diese finden dort ihre Grenzen, wo die Informations- und Kommunikationstechnik Informationen nicht in ausreichendem Maße zur Verfügung stellen können. (Frese /Organisation/ S. 153) Auch aus dieser Argumentation lässt sich der Vorteil einer Unternehmens-IT erkennen, die ortsunabhängig eingesetzt werden kann.

ponenten) sowie ihre Kommunikation als auch die Preisgabe ihrer Eigenschaften zu ihrer Umgebung. Alle diese Aspekte werden durch eine Softwarearchitektur festgelegt, weshalb Ersetzbarkeit zur Bewertung einer Softwarearchitektur herangezogen wird.

Einstufung der Gewichtung des Qualitätsattributes

Weil die Portabilität einer Software entscheidend von der zugrunde liegenden Softwarearchitektur abhängt[528] und für eine heutige Unternehmens-IT zur Grundanforderung werden kann, wird das Gewicht der Portabilität zur Bewertung einer Softwarearchitektur als hoch eingestuft.

3.1.4 Geschäftliche Qualitätsattribute

Anforderungen lassen sich in Produkt-, Unternehmens- und externe Anforderungen untergliedern.[529] *Produktanforderungen* sind Anforderungen, die sowohl aus technischer als auch aus fachlicher Sicht an ein Softwareprodukt gestellt werden. Sie umfassen z. B. Benutzbarkeits-, Effizienz-, Zuverlässigkeits- und Portierbarkeitsanforderungen.[530] *Unternehmensanforderungen* und *externe Anforderungen* sind dagegen Anforderungen, die aus der Sicht des Unternehmens und der externen Tätigkeit an eine Software gestellt werden. Unternehmensanforderungen beinhalten z. B. Liefer-, Umsetzungs- und Vorgehensanforderungen. Externe Anforderungen umfassen z. B. Kompatibilitäts-, ethische und rechtliche Anforderungen.

Unternehmen können heute nicht mehr auf den Einsatz von IT verzichten.[531] Darüber hinaus wird IT nicht mehr nur zur Effizienzsteigerung eingesetzt, sondern ermöglicht es, bisher unbekannte Produktvarianten zu erstellen, zu verwenden, zu vertreiben oder auch Produkte zu veredeln. Hierunter fällt z. B. die Schaffung neuer Beschaffungs-, Absatz- oder Kommunikationskanäle. Dadurch, dass Investitionen wegen der neuen Einsatzmöglichkeiten immer höher und langfristiger werden, wird die Wichtigkeit der Beachtung von *Unternehmensanforderungen und externe Anforderungen* an eine Un-

---

[528] Vgl. Raasch /Systementwicklung/ S. 34.

[529] Vgl. zum folgenden Absatz Sommerville /Software Engineering/ S. 109-114.

[530] Vgl. Kapitel 3.1.3.

[531] Vgl. Kapitel 1.1.2.

ternehmens-IT immer größer. Unternehmensanforderungen und externe Anforderungen werden in dieser Arbeit zur Bewertung einer Softwarearchitektur zu geschäftlichen Anforderungen zusammengefasst, die im Qualitätsmodell als ‚geschäftliche Qualitätsattribute' einfließen.

Diese Arbeit beschränkt sich auf drei geschäftliche Qualitätsattribute, die an eine Unternehmens-IT gestellt werden. Kapitel 3.1.4.1 beschreibt die Herleitung dieser Qualitätsattribute. Anschließend werden sie in den Kapiteln 3.1.4.2 – 3.1.4.4 vorgestellt.

3.1.4.1 Herleitung der geschäftlichen Qualitätsattribute

Es konnte weder ein anerkannter Standard noch eine zusammenführende Arbeit für geschäftliche Qualitätsattribute einer Unternehmens-IT gefunden werden. Deswegen musste, wie in Kapitel 3.1.1 dargestellt, zur Herleitung eines Qualitätsmodells für geschäftliche Qualitätsattribute eine alternative Vorgehensweise zu den technisch-fachlichen Qualitätsattributen angewendet werden.

Die für diese Arbeit identifizierten Qualitätsattribute zur Bewertung einer Softwarearchitektur sind: *Strategieunterstützung*, *Nachhaltigkeit* und *Integriertheit*. Sie konnten als Ergebnis einer Literaturstudie erarbeitet werden.

Das Qualitätsmodell der geschäftlichen Qualitätsattribute ist in Abb. 3-3 dargestellt. Die Herleitung der Qualitätsattribute werden in den folgenden Kapiteln 3.1.4.1.1 bis 3.1.4.1.3 vorgestellt. Anschließend werden die Qualitätsattribute, ihre Arbeitsdefinition sowie ihre Signifikanz in den Kapiteln 3.1.4.2 bis 3.1.4.4 erläutert.

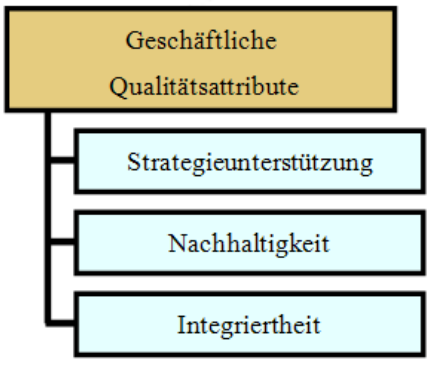

Abb. 3-3: Geschäftliche Qualitätsattribute

3.1.4.1.1 Herleitung der Strategieunterstützung

Die Strategieunterstützung zur Bewertung der Wirtschaftlichkeit einer Softwarearchitektur wird z. B. von Aier und Schönherr, Krüger und Laudon gefordert.

Aier und Schönherr formulieren in ihrer Arbeit die Anforderung, dass Informationssysteme bereitstehen müssen, die die strategische Ausrichtung eines Unternehmens unterstützen können.[532] Die Autoren stellen dar, dass durch moderne Technologien neue strategische Gestaltungsmöglichkeiten entstehen können.[533] Ein Beispiel ist die Strategie der Kostenführerschaft bei sog. ‚Billig-Airlines' wie z. B. Ryan Air. Sie konnte unter anderem durch die Nutzung neuer Vertriebswege über das Internet enormes Sparpotential erreichen. Aier und Schönherr zeigen auch auf, dass die Diskussion einer Verbindung strategischer Unternehmensziele mit technischen Infrastrukturen in der Wirtschaftsinformatik schon seit einigen Jahren geführt wird.[534]

Krüger[535] schreibt in seiner Arbeit über den Wandel von Unternehmen, dass der Prozess des Managements permanenten Wandels mit der operativen und strategischen Führung zu verzahnen ist.[536] Zu der operativen und strategischen Führung zählt nach Krüger auch die Führung der Planungs- und Informationssysteme.[537] Strategien ändern sich aus verschiedenen Gründen im Laufe der Zeit.[538] Der Prozess permanenten Wandels nach Krüger kann in diesem Kontext als Auslöser für Strategieänderung (stellvertretend für alle anderen) angesehen werden. Deswegen ist es für Softwarearchitekturen wichtig, einerseits den Wandel von Strategien zu verfolgen, weil sich Strategien nicht nur ändern, sondern auch auf Prozesse innerhalb eines Unternehmens Einfluss haben können.[539] Andererseits ist es für Softwarearchitekturen wichtig, verfolgte Strategien unter-

---

[532] Vgl. Aier, Schönherr /Flexibilisierung/ S. 21-22.

[533] Vgl. Aier, Schönherr /Flexibilisierung/ S. 8.

[534] Vgl. Aier, Schönherr /Flexibilisierung/ S. 45. Aier und Schönherr verweisen bei ihren Ausführungen auf andere Autoren wie Krcmar, Zachman, Scheer, Wall und Pohland. Vgl. Aier, Schönherr /Flexibilisierung/ S. 45-49.

[535] Krüger /Management/ S. 227-249.

[536] Vgl. Krüger /Management/ S. 239-240.

[537] Vgl. Krüger /Management/ S. 240.

[538] Vgl. Macharzina /Unternehmensführung/ S. 198-202.

[539] Vgl. Macharzina /Unternehmensführung/ S. 198-199.

stützen zu können, weil die Unternehmens-IT eines Unternehmens und die Prozesse eines Unternehmens unterstützen sollen.[540]

Auch Laudon[541] beschreibt indirekt die Herausforderung, dass Informationssysteme und ihre Infrastrukturen in der Lage sein sollten, den Strategien eines Unternehmens zu folgen. Laudon stellt in seinem Buch folgende Herausforderungen an das Management auf:[542]

- Strategien aufstellen, die sich Informationstechnologien zunutze machen können,
- Verständnis dafür aufbauen, welche Anforderung die Globalisierung an das Geschäft und an die Informationssysteme stellt,
- der Aufbau einer Informationssystemarchitektur und –infrastruktur, die die Ziele des Unternehmens unterstützen kann,
- die Bewertung von Informationssysteminvestitionen und
- Verantwortlichkeiten für Informationssysteme übernehmen.

Laudon stellt explizit die Forderung auf, dass das Management die Bedürfnisse und Fähigkeiten der Unternehmens-IT verstehen und einsetzten muss, um aktuellen Herausforderungen zu begegnen. Indirekt stellen sich weitere Herausforderungen an die Unternehmens-IT: Sie soll (1) durch das Management besser verstanden werden und (2) so eingesetzt werden, dass eine Positionierung und Befähigung erreicht wird, um die aufgestellten Herausforderungen unterstützten zu können. Auch Laudon stellt die Forderung auf, dass die Unternehmens-IT und explizit auch die zugrunde liegende Softwarearchitektur Strategien im Unternehmensumfeld unterstützen.

3.1.4.1.2 Herleitung der Nachhaltigkeit

Nachhaltigkeit zur Bewertung der Wirtschaftlichkeit einer Softwarearchitektur wird u. a. von Gronau und Krcmar gefordert.

---

[540] Vgl. Kapitel 2.2.5.1.

[541] Laudon /Management/.

[542] Vgl. Laudon /Management/ S. 26-29.

Gronau[543] beschreibt in seiner Arbeit ausführlich die Anforderung der Nachhaltigkeit an Informationssystemarchitekturen[544] und geht in seinen Ausführungen sehr sorgfältig vor. Die Kriterien nach Gronau gehen deswegen zum großen Teil in die Qualitätsbewertung einer Softwarearchitektur ein und stellen die entsprechenden Unterqualitätsattribute der Nachhaltigkeit. Gronau bezieht sich in seiner Arbeit auf Grundlagen nach Huber[545], der die Nachhaltigkeit im Kontext von Ökologie und Ökonomie untersuchte.

Krcmar stellt fest, dass die Bedeutung der Softwarearchitektur neben der strategischen Relevanz und den unternehmensweiten Auswirkungen auch aus der Langfristigkeit der Aktivitäten bei der Nutzung von Informations- und Kommunikationstechnologien resultiert.[546] Eine Softwarearchitektur beeinflusst die Möglichkeiten der weiteren Entwicklung eines Unternehmens. Eine Einschränkung stellt oftmals die mögliche Geschwindigkeit einer Weiterentwicklung dar, weil die Fortentwicklung einer Unternehmens-IT langsamer als erwartet geschieht.

3.1.4.1.3 Herleitung der Integriertheit

Integriertheit zur Bewertung der Wirtschaftlichkeit einer Softwarearchitektur wird von Linß[547] gefordert. Integriertheit verlangt auch Gronau[548] in seiner Arbeit zur Nachhaltigkeit von Informationssystemarchitekturen.

Linß entwickelt in seiner Arbeit ein Vorgehensmodell, das die Nutzenbetrachtung verschiedener Integrationsformen ermöglicht.[549] Mit der Begründung, dass Kosten, die mit einer Integration verbunden sind, im Allgemeinen leichter als der Nutzen zu ermitteln sind, geht Linß allerdings nicht explizit auf Kosten unterschiedlicher Integrationsformen ein.[550] Linß identifiziert vielfältige Literatur zum Thema Nutzen- bzw. Wirtschaftlichkeitsanalyse und analysiert ausgewählte Methoden auf Nutzeffekte, die Integrationsas-

---

[543] Gronau /Wandlungsfähige Informationssystemarchitekturen/.

[544] Vgl. Gronau /Wandlungsfähige Informationssystemarchitekturen/ S. 205-228.

[545] Huber /Nachhaltige Entwicklung/ S. 31-46.

[546] Vgl. zum folgenden Absatz Krcmar /Bedeutung/ S. 401.

[547] Linß /Nutzeffekte/.

[548] Gronau /Wandlungsfähige Informationssystemarchitekturen/.

[549] Vgl. Linß /Nutzeffekte/ S. 63-68.

[550] Vgl. Linß /Nutzeffekte/ S. 1.

pekte einer Unternehmens-IT betreffen.[551] Linß stellt ein Vorgehensmodell zur Nutzenbewertung auf, das sich in unterschiedlichen Situationen, für verschiedene Branchen, unter Beachtung unterschiedlicher Ziele zur Nutzenbewertung eignet. Je nach Durchführung kann ein unterschiedliches Ergebnis, nämlich die Bewertung einer konkreten Unternehmens-IT oder die Bewertung einer Integrationsform, erreicht werden.[552] In dieser Arbeit werden die zugrunde gelegten Bewertungskriterien nach Linß herausgearbeitet und in ein Qualitätsmodell als Qualitätsattribute überführt.

Gronau stellt in seiner Arbeit zur Nachhaltigkeit von Informationssystemarchitekturen auch das Kriterium der Integriertheit (als Unterqualitätsattribut der Nachhaltigkeit) auf.[553] Um den Begriff der Nachhaltigkeit für diese Arbeit jedoch nicht zu allgemein auszulegen, wird der Auffassung von Linß gefolgt, dass die Integriertheit als eigenständiges Bewertungskriterium zur Nutzenbewertung von Softwarearchitekturen verwendet werden kann.[554]

Das geschäftliche Qualitätsattribut Integriertheit wird gegenüber des technisch-fachlichen Qualitätsattributs Interoperabilität[555] abgegrenzt. Die Integriertheit bildet die Fähigkeit einer Softwarearchitektur ab, sich in die unternehmensinternen Prozesse und Systeme eines Unternehmens integrieren zu können. Interoperabilität dagegen bildet die Fähigkeit einer Softwarearchitektur ab, mit unternehmensexternen Systemen kommunizieren zu können.

3.1.4.2 Strategieunterstützung

Eine integrierte Betrachtung strategischer Gesichtspunkte mit der Softwarearchitektur einer Unternehmens-IT[556] wurde von Krcmar schon 1990 mit der Begründung gefor-

---

[551] Vgl. Linß /Nutzeffekte/ S. 47-62.

[552] Vgl. Linß /Nutzeffekte/ S. 111-114.

[553] Vgl. Gronau /Wandlungsfähige Informationssystemarchitekturen/ S. 213.

[554] Im anderen Fall, dass man Integriertheit zur Nachhaltigkeit einem allgemeinen Verständnis folgend unterordnet, müssten ansonsten auch weitere Qualitätsattribute, die Einfluss auf zukünftige Ereignisse oder Anforderungen haben, wie die Strategieunterstützung, Wandlungsfähigkeit oder Wiederverwendbarkeit, als Unterqualitätsattribute der Nachhaltigkeit aufgefasst werden. Stattdessen werden in dieser Arbeit nur direkte Vergleichskriterien wie die strukturelle Analogie und Autopoiesis herangezogen. Vgl. Kapitel 3.1.4.3.1.

[555] Vgl. Kapitel 3.1.3.2.2.

[556] Krcmar spricht hier von der Informationssystem-Architektur aus ganzheitlicher Sicht.

dert, dass sich ansonsten unzweckmäßige Systemabgrenzungen ergeben.[557] Denn die Unternehmens-IT ist, wie schon mehrfach festgestellt, ein unterstützendes System für das Unternehmen.[558] Als ein solches unterstützendes System müssen sich die Unternehmens-IT und die zugrunde gelegte Softwarearchitektur den Änderungen eines Unternehmens anpassen können; und zwar sowohl bei Struktur als auch Strategie.[559] Weil Unternehmen permanentem Wandel ausgesetzt sind[560] und sich Unternehmensstrategien entsprechend ändern können, muss die Unternehmens-IT und die zugrunde gelegte Softwarearchitektur insbesondere auch den Wandel von Unternehmensstrategien unterstützen können.

3.1.4.2.1 Erläuterung der Strategieunterstützung

Der Begriff der Strategie im Unternehmenskontext wird unterschiedlich aufgefasst.[561] Erstens kann unter der Strategie ein umfassendes Maßnahmenbündel verstanden werden, das rational geplant werden kann und dessen Formulierung vor der Maßnahmenrealisierung erfolgt.[562] Zweitens kann Strategie auch als Grundmuster im Strom der Entscheidungen oder Aktivitäten eines Unternehmens angesehen werden.[563] Im zweiten Fall wird das Handeln selbst zur Unternehmensstrategie, entgegen dem Verständnis, dass das Handeln aus der verfolgten Strategie resultiert.[564]

In dieser Arbeit wird nicht zwischen diesen Definitionen einer Strategie unterschieden, sondern vielmehr auf Ausprägungen entsprechender Strategien eingegangen. Eine Unterscheidung des Strategiebegriffes auf dieser Ebene ist für diese Arbeit nicht notwendig. Für die Bewertung ist es unerheblich, ob eine Softwarearchitektur eine Strategie gemäß der einen oder der anderen Definition unterstützt. Für das Qualitätsmodell dieser Arbeit ist es vielmehr ausreichend, dass unabhängig von der Definition eine oder meh-

---

[557] Vgl. Krcmar /Bedeutung/ S. 399.

[558] Vgl. Kapitel 2.2.5.1.

[559] Vgl. Oxman, Smith /Structual Change/ S. 77.

[560] Vgl. Kapitel 1.1.

[561] Vgl. zur Darstellung der unterschiedlichen Auffassungen Macharzina /Unternehmensführung/ S. 197-202.

[562] Vgl. Macharzina /Unternehmensführung/ S. 197-199.

[563] Vgl. Macharzina /Unternehmensführung/ S. 199-202.

[564] Vgl. Macharzina /Unternehmensführung/ S. 200.

rere Strategien vorliegen und verfolgt werden. Für eine Unternehmens-IT stellt sich die Herausforderung, den entsprechenden Strategien folgen und diese unterstützen zu können.

Strategieunterstützung als Qualitätsattribut wird in dieser Arbeit auf zwei Ebenen untersucht. Erstens, inwieweit eine Softwarearchitektur mögliche Strategien unterstützen kann, und zweitens, ob und wie gut eine Softwarearchitektur den Übergang von einer Strategie zu einer anderen vollziehen kann. Zur ersten Ebene findet in dieser Arbeit eine Fokussierung auf die nach Macharzina wichtigen Strategietypen statt:[565] Gesamtunternehmensstrategien (Corporate Strategies), Wettbewerbsstrategien (Competitive Strategies), Geschäftsbereichsstrategien (Business Strategies) und Funktionsbereichsstrategien (Functional Strategies). Die zweite Ebene wird direkt anhand der Unterstützung des Strategiewandels untersucht:

- Unterstützung von Gesamtunternehmensstrategien

  Nach Macharzina stehen im Mittelpunkt der Formulierung einer Gesamtunternehmensstrategie Produkt-Markt-Strategien.[566] Diese legen fest, welche Produkte bzw. Dienstleistungsbereiche auf welchen Märkten angeboten werden sollen. Die wichtigsten Ausprägungen dieser Strategien sind Marktdurchdringung, Produkt- und Markterweiterung sowie Diversifikation.[567] Weil die Softwarearchitektur einer Unternehmens-IT das Bindeglied zwischen Gesamtunternehmensstrategie und ihre Umsetzung in die Nutzung der Informations- und Kommunikationstechnik darstellt,[568] fließt die Bewertung der Unterstützung von Gesamtunternehmensstrategien ein, indem die Fähigkeit einer Softwarearchitektur bewertet wird, Produkt-Markt-Strategien unterstützen zu können.

  Kieser und Walgenbach identifizieren ‚Internationalisierung' als neuere Ausprägung einer Gesamtunternehmensstrategie.[569] Durch die Verfolgung einer solchen Strategie versuchen Unternehmen international zu agieren, um sich den Zugang zu Ressourcen zu sichern, um Skalen- und Verbundeffekte zu realisieren, um unterschiedliche Aus-

---

[565] Vgl. Macharzina /Unternehmensführung/ S. 203-209.

[566] Vgl. zum folgenden Absatz Macharzina /Unternehmensführung/ S. 203-205.

[567] Vgl. Macharzina /Unternehmensführung/ S. 250-253.

[568] Vgl. Krcmar /Bedeutung/ S. 401.

[569] Vgl. zum folgenden Absatz Kieser, Walgenbach /Organisation/ S. 284-285.

prägungen der Faktorenkosten verschiedener Länder ausnutzen und um erweiterte Möglichkeiten des Risikomanagements nutzen zu können.

- Unterstützung von Wettbewerbsstrategien
  Eine wichtige Basisarbeit zu Wettbewerbsstrategien hat Porter geleistet.[570] Porter identifiziert drei grundsätzliche Wettbewerbsstrategien:[571] Kostenführerschaft, (Produkt-)Differenzierung und Nischenstrategie (vgl. Abb. 3-4). Die Bewertung der ‚Unterstützung von Wettbewerbsstrategien' beurteilt die Fähigkeit einer Softwarearchitektur, alle diese unterstützen zu können.

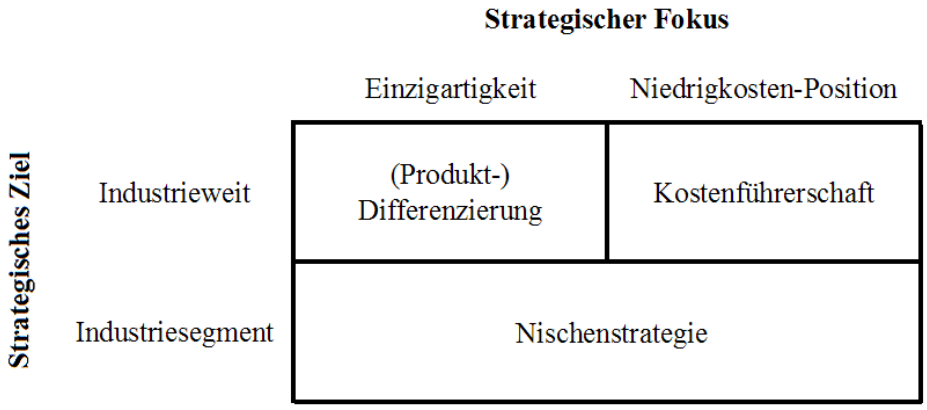

Abb. 3-4: Wettbewerbsstrategien nach Porter[572]

- Unterstützung von Geschäftsbereichs- und Funktionsbereichsstrategien
  Geschäftsbereichsstrategien sind solche, die auf die Produkt-Markt-Strategie eines einzelnen Geschäftsbereiches ausgerichtet sind.[573] Synonym dazu sind Funktionsbereichsstrategien solche, die auf die Produkt-Markt-Strategie eines einzelnen Funktionsbereiches ausgerichtet sind und zusätzlich mit Geschäftsbereichsstrategien abgestimmt sein müssen.[574] Geschäftsbereichsstrategien leiten sich deduktiv aus Gesamtunternehmensstrategien ab und Funktionsbereichsstrategien entsprechend aus Geschäftsbereichsstrategien.[575] Beide sind Strategietypen, die sich aus Gesamtunternehmensstrategien ableiten. Die Anforderungen, die sich an eine Unternehmens-IT entsprechend ergeben würden, können aus den Anforderungen einer Gesamtunternehmensstrategie abgeleitet werden. Infolgedessen ist die Bewertung der Unterstüt-

---

[570] Vgl. Macharzina /Unternehmensführung/ S. 205.

[571] Vgl. Porter /Advantage/ S. 11-16; Porter /Strategy/ S. 34-40.

[572] Porter /Strategy/ S. 39.

zung von Geschäftsbereichs- und Funktionsbereichsstrategien eingeschlossen in der von Gesamtunternehmensstrategien. Eine Doppelbewertung wird also vermieden.[576]

- Unterstützung des Strategiewandels
  Strategien ändern sich im Laufe der Zeit.[577] Weil eine Unternehmens-IT das Unternehmen unterstützen soll,[578] muss eine Softwarearchitektur den Wandel von Strategien unterstützen können. Die Fähigkeit, den Strategiewandel zu unterstützen ist relevant für Softwarearchitekturen, weil Unternehmen ihre Strategien ändern.[579]

Aus diesen Überlegungen leitet sich die Definition des folgenden Kapitels ab.

3.1.4.2.2 Arbeitsdefinition der Strategieunterstützung

Strategieunterstützung bedeutet die Fähigkeit einer Software, dem Unternehmensgestalter die Freiheit zu geben, Unternehmensstrategien formulieren und verfolgen zu können, ohne dass Einschränkungen durch die Software bestehen.

- Unterstützung von Gesamtunternehmensstrategien
  Unterstützung von Gesamtunternehmensstrategien ist die Fähigkeit einer Softwarearchitektur, Produkt-Markt-Strategien unterstützen zu können.
- Unterstützung von Wettbewerbsstrategien
  Unterstützung von Wettbewerbsstrategien ist die Fähigkeit einer Softwarearchitektur, eine Strategie der Kostenführerschaft, der (Produkt-)Differenzierung und eine Nischenstrategie unterstützen zu können.
- Unterstützung des Strategiewandels
  Unterstützung des Strategiewandels ist die Fähigkeit einer Softwarearchitektur, den Strategiewandel eines Unternehmens zu unterstützen.

---

[573] Vgl. Macharzina /Unternehmensführung/ S. 208-209.

[574] Vgl. Macharzina /Unternehmensführung/ S. 209.

[575] Vgl. Macharzina /Unternehmensführung/ S. 208-209.

[576] Auch bei der Betrachtung zur Strategie eines Unternehmens bezieht sich Mellis ‚nur' auf die beiden Strategiearten ‚Unternehmensstrategie' und ‚Wettbewerbsstrategie'. Vgl. Mellis /Projektmanagement/ S. 31.

[577] Vgl. Macharzina /Unternehmensführung/ S. 207-208.

[578] Vgl. Kapitel 2.2.5.1.

[579] Vgl. Macharzina /Unternehmensführung/ S. 199.

3.1.4.2.3 Signifikanz der Strategieunterstützung

Vor allem für große Unternehmen ist die Verfolgung einer Strategie notwendig für die unternehmerische Ausrichtung der Tätigkeiten,[580] und große Unternehmen müssen heute Informationssysteme einsetzen, um wettbewerbsfähig bleiben zu können.[581] Weil eine Unternehmens-IT auf einer Softwarearchitektur aufbaut, ist die Betrachtung der Strategieunterstützung für Softwarearchitekturen wichtig und umso bedeutsamer, je größer ein Unternehmen ist.

Die Relevanz wird verstärkt, weil häufig eine weitere Steigerung operativer Vorteile gegenüber Mitbewerbern nicht mehr realisierbar ist, und die strategische Positionierung gegenüber Mitbewerbern umso wichtiger wird.[582] Des Weiteren wird die Relevanz durch Autoren wie Porter aufgezeigt: Für den Bereich der Wettbewerbsstrategien hebt er hervor, dass Technologie großen Einfluss auf Kosten und Differenzierung haben kann.[583]

Signifikanz der Unterqualitätsattribute

- Unterstützung von Gesamtunternehmensstrategien
  Gesamtunternehmensstrategien sollen das Handeln eines gesamten Unternehmens ausrichten.[584] Die Formulierung von Gesamtunternehmensstrategien dient der Steuerung eines Unternehmens. Durch die starke Durchdringung arbeitsteiliger Prozesse in Unternehmen erhält die Möglichkeit der Steuerung des Unternehmens durch die Strategieformulierung einen hohen Stellenwert für Unternehmen und Manager.[585] Für die Unternehmens-IT (und die zugrunde liegende Softwarearchitektur) wird die Unterstützung der Gesamtunternehmensstrategie relevant, weil die Unternehmens-IT Geschäftsprozesse unterstützen soll, die an einer Gesamtunternehmensstrategie ausgerichtet wurden.

---

[580] Dies wird als Grundaussage in Macharzina /Unternehmensführung/ S. 203 deutlich. Macharzina geht darauf ein, dass in der ersten Hälfte des 19. Jahrhunderts Strategiefragen noch nebensächliche Bedeutung hatten, und der Wille des Einzelunternehmers die Grundzüge des Handelns eines Unternehmens darstellte.

[581] Vgl. 1.1.2.

[582] Vgl. Porter /Internet/ S. 70-72.

[583] Vgl. Porter /Advantage/ S. 169-171.

[584] Vgl. Macharzina /Unternehmensführung/ S. 203-204.

- Unterstützung von Wettbewerbsstrategien

  Wettbewerb wird als Keimzelle unternehmerischen Erfolgs oder Misserfolgs angesehen.[586] Um im Wettbewerb bestehen zu können, muss die Handlung eines Unternehmens durch die Formulierung von Wettbewerbsstrategien zielgerichtet erfolgen.[587] Die entsprechenden Prozesse werden durch die Unternehmens-IT unterstützt. Eine zugrunde liegende Softwarearchitektur muss folglich in der Lage sein, Wettbewerbsstrategien unterstützen zu können.

- Unterstützung des Strategiewandels

  Unternehmen sind heute hohem Wandlungsdruck ausgesetzt.[588] Insbesondere sind Unternehmen betroffen, die in turbulenten Umfeldern[589] agieren, die stark wachsen und oft geänderten Marktbedingungen gegenüber stehen. Folglich sind und werden Unternehmen immer häufiger gezwungen, ihre strategische Ausrichtung neuen Marktanforderungen anzupassen.[590] Eine Softwarearchitektur muss deshalb in der Lage sein, den Strategiewandel eines Unternehmens zu unterstützen. Besondere Relevanz erhält das Unterqualitätsattribut durch Firmenübernahmen.[591] Gerade hierdurch werden Unternehmen gezwungen, ihre bisherigen Strategien anzupassen oder zu erweitern. Auch wird die IT-Integration nach Firmenzusammenschlüssen als eine der wichtigsten Aktivitäten angesehen, um Firmenzusammenschlüsse erfolgreich umsetzen zu können.[592]

Einstufung der Gewichtung des Qualitätsattributes

Wie dargestellt ist die Strategieunterstützung für Softwarearchitekturen von großer Bedeutung. Aus diesem Grund fließt die Bewertung der Strategieunterstützung mit hohem Gewicht in die Bewertung einer Softwarearchitektur ein.

---

585 Vgl. Macharzina /Unternehmensführung/ S. 197-198.

586 Vgl. Porter /Advantage/ S. 1.

587 Vgl. Macharzina /Unternehmensführung/ S. 205-206.

588 Vgl. Kapitel 1.1.1.

589 Vgl. Mellis /Turbulent Times/ S. 278.

590 Vgl. Porter /Strategy/ S. 3-6.

591 Vgl. Aier, Schönherr /Flexibilisierung/ S. 13.

592 Vgl. Trapp, Otto /Einsatzmöglichkeiten/; Epstein /Drivers/ S. 174-175; Keller /EAI/ S. 13-14; Kaib /EAI/ S. 43.

3.1.4.3 Nachhaltigkeit

Nachhaltigkeit im klassischen Sinne bedeutet die Betrachtung der Ökonomie und Ökologie als eine integrierte Systemkomponente.[593] In dieser Arbeit wird der Begriff Nachhaltigkeit jedoch vollkommen losgelöst von der Ökologie verstanden und stattdessen dem langfristigen Oberziel einer Unternehmensführung angepasst.

3.1.4.3.1 Erläuterung der Nachhaltigkeit

Das Oberziel einer Unternehmensführung ist die Erhaltung und erfolgreiche Weiterentwicklung eines Unternehmens zur Erfüllung der Individualziele aller an dem Unternehmen interessierten Gruppen.[594] Übertragen auf die Nachhaltigkeit der Unternehmens-IT bedeutet dies die Betrachtung eines Unternehmens und der entsprechenden Unternehmens-IT als eine integrierte Systemkomponente, die erhalten und weiterentwickelt werden soll. Bei der Bewertung der Nachhaltigkeit einer Unternehmens-IT werden organisatorische Veränderungen der Vergangenheit, Gegenwart und Zukunft sowie die durch sie erforderliche Anpassungsfähigkeit der vorhandenen und zu erschaffenden Softwarekomponenten betrachtet.[595]

Gronau stellt ein Bewertungsschema zur Bewertung von Softwarearchitekturen auf (er nennt diese Informationssystemarchitekturen) und belegt die Anwendbarkeit anhand von Beispielen.[596] Das sorgfältig ausgearbeitete Bewertungsschema von Gronau[597] wird zur Bewertung der Nachhaltigkeit einer Softwarearchitektur für diese Arbeit übernommen. Es gestaltet sich nach folgenden Qualitätsattributen:[598]

---

[593] Vgl. Huber /Nachhaltige Entwicklung/ S. 32.

[594] Vgl. Hahn, Hungenberg /Controllingkonzepte/ S. 13.

[595] Vgl. Gronau /Wandlungsfähige Informationssystemarchitekturen/ S. 213.

[596] Vgl. Gronau /Wandlungsfähige Informationssystemarchitekturen/ S. 222-228, sowie für Beispiele S. 228-232.

[597] Gronau leitet Nachhaltigkeitsanforderungen ab und betrachtet dabei klassische Arbeiten zur Nachhaltigkeit im ökologischen Sinn. Er bewertet die Einsatzfähigkeit der Ansätze klassischer und neuerer Arbeiten und kommt zu mehreren Qualitätskriterien. Während seiner Untersuchungen erkennt er, dass die klassischen – auf die Ökologie abgestimmten – Strategien (Konsistenz, Effizienz, Suffizienz) nicht geeignet sind, um nachhaltige Informationssystemarchitekturen bewerten zu können. Vgl. Gronau /Wandlungsfähige Informationssystemarchitekturen/ S. 221-222.

[598] Vgl. Gronau /Wandlungsfähige Informationssystemarchitekturen/ S. 218-219.

- Strukturanalogie

  Gronau identifiziert vier grundlegende Reorganisationsansätze[599] und stellt für diese Prinzipien auf, die die entsprechenden Reorganisationsansätze unterstützen. Diese Prinzipien sind das Prinzip der Dezentralität, das Prinzip der Flexibilität und Dynamik[600] und das Prinzip der strukturellen Analogie.[601] Das Prinzip der *Dezentralität* betrifft Strukturen und Prozesse auf zeitlicher und räumlicher Ebene. Das Prinzip der *Flexibilität und Dynamik* betrifft die Schaffung und Aufrechterhaltung eines Flusses durch Prozesse und Veränderungen der unternehmensinternen und unternehmensübergreifenden Strukturen (wie z. B. Allianzen). Das Prinzip der *strukturellen Analogie* betrifft das Merkmal, dass sich eine Unternehmens-IT harmonisch an den jeweils vorherrschenden Organisationstyp anpassen kann.[602]

- Autopoiesis[603]

  Das Konzept eines offenen, evolutionären und zumindest teilweise autonomen Systems wird autopoietisches System genannt.[604] Anforderungen zur Architekturbewertung, die sich nach Gronau daraus ergeben, sind partielle Autonomie und Selbstorganisation.[605] Partielle Autonomie fordert, dass Softwarearchitekturen die

---

599 Dies sind die Reorganisationsansätze ‚Segmentierung', ‚Prozessorientierung', ‚kontinuierliche Reorganisation' sowie ‚Auflösung von Unternehmensgrenzen'. Vgl. Gronau /Wandlungsfähige Informationssystemarchitekturen/ S. 60-64.

600 Flexibilität wurde in dieser Arbeit in den Ausführungen zur Wandlungsfähigkeit (vgl. Kapitel 3.1.3.3.1) erwähnt. Für die Bewertung einer Softwarearchitektur findet keine ‚Doppelbewertung' der Flexibilität einer Softwarearchitektur statt. Diese Bewertung erfolgt nur bei den technisch-fachlichen Qualitätsattributen. Diese getroffene Bewertung der Flexibilität wird für die Bewertung der Nachhaltigkeit einer Softwarearchitektur berücksichtigt und fließt somit als Bewertungshilfe ein. Allerdings erfährt die Gewichtung der Wandlungsfähigkeit durch die Verwendung des Bewertungsergebnisses bei der Bewertung der Nachhaltigkeit indirekt ein höheres Gewicht. Weil allerdings auch eine Abhängigkeit zur Nachhaltigkeit besteht, ist diese indirekte und im Vergleich zu anderen Qualitätsattributen höhere Gewichtung gerechtfertigt.

601 Vgl. zum folgenden Absatz Gronau /Wandlungsfähige Informationssystemarchitekturen/ S. 216-217.

602 Den Nachweis, dass eine solche Anpassung zwischen Organisationsentwicklung und Unternehmens-IT stattfindet, erbringt Gronau auch in seiner Arbeit. Vg. Gronau /Wandlungsfähige Informationssystemarchitekturen/ S. 217-218.

603 Vgl. Gronau /Wandlungsfähige Informationssystemarchitekturen/ S. 209.

604 Vgl. Gronau /Wandlungsfähige Informationssystemarchitekturen/ S. 219-220.

605 Vgl. zum folgenden Absatz Gronau /Wandlungsfähige Informationssystemarchitekturen/ S.219-221. Gronau stellt in seinem Modell zur Bewertung neben den beiden Attributen partielle Autonomie und Selbstorganisation noch das Attribut ‚Selbstähnlichkeit' auf. (Vgl. Gronau /Wandlungsfähige Informationssystemarchitekturen/ S.221) Allerdings wird das Attribut durch Gronau nicht erläutert, und erst zum Ende seiner Ausführungen zur Autopoiesis als Bewertungsattribut seines Anforderungsmodells aufgestellt. In den Augen des Autors dieses Beitrags ist Selbstähnlichkeit als Qualitätsattribut nicht tragfähig, da die Bedeutung des Begriffes implizit immer erfüllt sein wird: Bewertungsobjekte sind sich nicht nur selbst ähnlich, sondern darüber hinaus sich selbst gleich/identisch. Die

Unternehmens-IT befähigt, ein Maximum an Aufgaben autonom abwickeln zu können. Selbstorganisation bedeutet, dass sich eine Unternehmens-IT und Teile einer Unternehmens-IT selbstständig verwalten können.

- Integration der Informationssystemarchitektur
  Integriertheit wird auch von Linß als Bewertungskriterium für Softwarearchitekturen identifiziert. In dieser Arbeit wird die Sicht von Linß vertreten und Integriertheit als eigenständiges Qualitätsattribut aufgenommen.[606] Aus diesem Grund wird an dieser Stelle auf die Ausführungen zur Integriertheit in Kapitel 3.1.4.4 verwiesen.

Aus diesen Überlegungen leitet sich die Definition des folgenden Kapitels ab.

3.1.4.3.2 Arbeitsdefinition der Nachhaltigkeit

Nachhaltig ist die Fähigkeit einer Unternehmens-IT, in mindestens ausreichendem Maße dauerhaft zukunftsfähig zu sein.[607] Nachhaltigkeit ist gegeben, wenn eine Unternehmens-IT strukturanaloge und autopoietische Merkmale aufweist.[608] Nachhaltigkeit als Qualitätsattribut dieser Arbeit definiert sich nur über die Unterqualitätsattribute:[609]

- Strukturanalogie und
- Autopoiesis.

3.1.4.3.3 Signifikanz der Nachhaltigkeit

Das Konzept der Nachhaltigkeit für die Bewertung der Wirtschaftlichkeit einer Unternehmens-IT ist als geschäftliches Qualitätsattribut relevant, weil erstens Unternehmen

---

Verwendung des Begriffes unter der Auffassung, dass andere Objekte einem Bewertungsobjekt ähnlich sein sollen, wird im Modell nach Gronau explizit durch die Bewertung der Strukturanalogie vorgenommen. In dieser Arbeit wird das von Gronau aufgestellt Attribut ‚Selbstähnlichkeit' deswegen nicht bewertet.

606 Vgl. Kapitel 3.1.4.1.3.

607 Vgl. Gronau /Wandlungsfähige Informationssystemarchitekturen/ S. 223.

608 [Anm. des Autors:] Im Gegensatz zu anderen Qualitätsattributen definiert sich die Nachhaltigkeit somit über ihre Unterqualitätsattribute und wird durch diese nicht „nur" detailliert. Andernfalls bestünde die Gefahr, dass sich ein zu breites Verständnis der Nachhaltigkeit etablieren könnte. Ein solches breiteres Verständnis müsste beispielsweise alle Nutzenaspekte von Qualitätsattributen mit zeitlich weiter Perspektive, wie Wandlungsfähigkeit, Strategieunterstützung, Wiederverwendbarkeit usw. umfassen.

609 Zur Erläuterung vgl. Kapitel 3.1.4.4.1.

Nachhaltigkeit von ihrer Unternehmens-IT fordern und zweitens eine Nutzenbewertung über den gesamten Lebenszyklus einer Unternehmens-IT durchgeführt werden muss.

Nach Gronau ist Nachhaltigkeit eine Anforderung, die das Unternehmen an ihre IT stellt und die für die Unternehmens-IT eine Notwendigkeit geworden ist.[610] Bei der Gestaltung der Unternehmens-IT sollen organisatorische Veränderungen der Vergangenheit, Gegenwart und Zukunft und die durch sie erforderliche Anpassungsfähigkeit der vorhandenen und zu erschaffenden Softwarekomponenten betrachtet werden. Ziel ist, dass das Informationssystem einer Organisation plus Organisation zusammen harmonieren und dass Informationssysteme an sich ändernde organisatorische Rahmenbedingungen anpassungsfähig sind.

Zum zweiten Punkt lässt sich beobachten, dass Unternehmen viel Geld in ihre Unternehmens-IT investieren.[611] Damit sich diese Investitionen rentieren müssen die Softwaresysteme einige Jahre nutzbar sein. In der Praxis zeigt sich, dass große Softwaresysteme zehn Jahre und länger in Betrieb sind (sog. Legacy-Systeme[612]). Die Fähigkeit einer Unternehmens-IT viele Jahre nutzbar zu sein wird durch die Nachhaltigkeit einer Softwarearchitektur unterstützt, weil sich die Nachhaltigkeit einer Softwarearchitektur auf die Nachhaltigkeit der auf ihr aufbauenden Unternehmens-IT auswirkt. Eine starke Erweiterung der zeitlichen Perspektive zur Nutzenbewertung ist dringend notwendig, weil sich Nutzen (wie auch Kosten) über den gesamten Lebenszyklus einer Software erstreckt. Gronau hebt zum Ansatz nachhaltiger Informationssystemarchitekturen hervor: „[Der] revolutionäre Charakter liegt in einer starken Erweiterung der zeitlichen Perspektive.“[613]

Signifikanz der Unterqualitätsattribute

- Strukturanalogie

  Die Relevanz der Strukturanalogie ergibt sich daraus, dass eine Unternehmens-IT ein Unternehmen bestmöglich unterstützen muss. Eine Unterstützung ist potentiell möglich, wenn die Struktur der Unternehmens-IT der Struktur des Unternehmens ähnelt.

---

[610] Vgl. zum folgenden Absatz Gronau /Wandlungsfähige Informationssystemarchitekturen/ S. 213-216.

[611] Vgl. zum folgenden Ansatz Sommerville /Sofware Engineering/ S. 589.

[612] Vgl. Sommerville /Software Engineering/ S. 589-606; Inmon, Zachman, Geiger /Data/ S. 33.

[613] Gronau /Wandlungsfähige Informationssystemarchitekturen/ S. 213.

Eine Abstimmung der Prozesse und Bedürfnisse zwischen der Unternehmens-IT und dem Unternehmen kann ohne unnötige strukturelle Anpassungen erfolgen.

- Autopoiesis
  Autopoietische Systeme besitzen die Fähigkeit, autonom handeln und sich selbst organisieren zu können. Sie können sich eigenständig den Änderungen der Umwelt und daraus resultierenden Anforderungen anpassen. Die Fähigkeit der Autopoiesis ist relevant, weil sich Unternehmen heute in einem Zustand permanenten Wandels befinden,[614] und autopoietische Systeme den Wandel des Unternehmens unterstützen können.

Einstufung der Gewichtung des Qualitätsattributes

Nachhaltigkeit ist für eine Unternehmens-IT und mithin auch für ihre Softwarearchitektur – wie aufgezeigt – für Unternehmen von hoher Wichtigkeit. Aus diesem Grund geht die Bewertung der Nachhaltigkeit mit hohem Gewicht in die Bewertung einer Softwarearchitektur ein.

3.1.4.4 Integriertheit

Integration ist die „Vereinigung von einzelnen Bestandteilen oder Komponenten zu einem System, bei der grundsätzlich gilt, dass die Summe der Elemente mehr ist als die einzelnen Elemente.“[615] Hierunter wird das Herstellen eines Ganzen verstanden, ohne dass die Gesamtheit bereits vorliegt.[616] Das Qualitätskriterium der Integriertheit wurde von Linß aufgestellt und untersucht.

3.1.4.4.1 Erläuterung der Integriertheit

Der Erfolg eines Unternehmens kann stark von dem Erfolg des jeweiligen geschäftlichen Ökosystems abhängen.[617] Zum Ökosystem eines Unternehmens gehören z. B. Lieferanten und Kunden. Iansiti und Levien haben herausgefunden, dass erfolgreiche Fir-

---

[614] Vgl. Kapitel 1.1.1.

[615] Schulze /Computer/ S. 1531.

[616] Vgl. Schulze /Computer/ S. 1532; Heinrich, Burgholzer /Systemplanung/ S. 149.

[617] Vgl. zum folgenden Absatz Iansiti, Levien /Strategy/ S. 69-70.

men Strategien verfolgt haben, die die Stärke ihres gesamten Ökosystems unterstützen. Sie haben sog. Plattformen erzeugt, die jedem Teilnehmer ihres Ökosystems zur Performanzverbesserung zur Verfügung stehen. Die Plattformen können sowohl Dienstleistungen bzw. Werkzeuge als auch Technologien darstellen. Nicht selten werden nur finanzielle Aspekte kurzfristiger Reichweite optimiert, statt die Gesundheit des Ökosystems eines Unternehmens zu erhalten und seine Stärken zu erhöhen.[618] Diese verfehlten Maßnahmen können das Ökosystem ggf. sogar zerstören.[619]

Darüber hinaus sind Informationsansprüche heute stark angestiegen und umfassen nicht nur die Kommunikation mit direkten Geschäftspartnern, sondern auch die Kommunikation mit Personen, zu denen keine absatzmarktbezogenen Beziehungen bestehen.[620] Durch ein gestiegenes Bedürfnis nach Information und nach Aktualität dieser Information müssen Informationsansprüche heute öfters, schneller und zielgruppenspezifischer bereitgestellt werden.

Das Ziel von Unternehmen ist folglich, das Ökosystem zu erhalten, die Leistungsfähigkeit des Ökosystems zu verbessern und an das Unternehmen gestellte Informationsansprüche anforderungsgerecht bedienen zu können. Um dieses Ziel zu erreichen soll eine Unternehmens-IT die entsprechenden Mechanismen der Konnektivität und Interoperabilität anbieten, damit dem Ökosystem eines Unternehmens entsprechende Dienstleistungen, Werkzeuge und Technologien zur Verfügung gestellt werden können.

Linß arbeitet das Qualitätskriterium Integriertheit sehr detailliert aus. Zur Aufstellung eines Modells zur Bewertung der Integriertheit einer Softwarearchitektur untersucht er sechs Arbeiten[621] zur Integration. Das Bewertungsmodell nach Linß (vgl. Abb. 3-5) wird zur Bewertung der Integriertheit verwendet.

Integrationsdimensionen nach Linß:[622]

---

[618] Vgl. Iansiti, Levien /Strategy/ S. 78.

[619] Vgl. Iansiti, Levien /Strategy/ S. 78.

[620] Vgl. zum folgenden Absatz Werder, Grundei, Talaulicar /Unternehmenskommunikation/ S. 397-398.

[621] Dies sind Arbeiten von Mertens, Scheer, Schumann, Heilmann, Hellmann und Horbach sowie Heinrich und Burgholzer aus den Jahren 1989–1993, die sich mit der Integration im Unternehmensumfeld und teilweise speziell mit der Unternehmens-IT beschäftigen. Vgl. Linß /Nutzeffekte/ S. 7-17.

[622] Vgl. zum folgenden Absatz Linß /Nutzeffekte/ S. 17-27.

- Integrationsgegenstand[623]

  Integrationsgegenstand einer Unternehmens-IT können Daten, Funktionen und Programme sein.[624] Eine Softwarearchitektur beeinflusst die Integrationsfähigkeit aller drei Gegenstände: Softwarearchitekturen geben Richtlinien zur Gestaltung von Softwarekomponenten. Diese Richtlinien umfassen sowohl die Datenstruktur, die Aufteilung von Funktionen auf Softwarekomponenten und sogar einzelne Programme (als Zusammensetzung von Softwarekomponenten). Auch die Integrationsfähigkeit einzelner Programme mit den dazugehörigen Daten und Funktionen werden durch eine unternehmensweite Softwarearchitektur beeinflusst, weil diese Gestaltungsrichtlinien für eine mögliche Integration in andere Bestandteile einer Unternehmens-IT definiert werden.

- Integrationsreichweite[625]

  Die Integrationsreichweite kann innerbetrieblich und zwischenbetrieblich betrachtet werden. Beide Sichtweisen haben als Bewertungsgegenstand die Fähigkeit einer Softwarearchitektur, entsprechende Gegenstände der jeweiligen Sichtweise integrieren zu können. Diese Fähigkeit beruht darauf, dass die Softwarekomponenten miteinander kommunizieren können. Inner- und zwischenbetriebliche Kommunikationsmöglichkeiten werden Softwarearchitekturen beeinflusst, da diese die Beziehungen der Softwarekomponenten zueinander regeln.[626] Die Integrationsreichweite ist folglich zur Bewertung einer Softwarearchitektur relevant.

- Integrationsrichtung[627]

  Integration kann nach Linß in horizontale und vertikale Integration unterschieden werden.[628] *Horizontale Integration* betrifft die Integration von Bereichen eines (oder mehrerer) Unternehmen(s) und kann nach der Anzahl der integrierten Bereiche und dem Grad der Integration bewertet werden. *Vertikale Integration* betrifft die Integration von Daten und Prozessen zur Leistungserstellung eines Unternehmens in Administrations- und Kontrollaufgaben. Die Integrationsrichtung betrifft sowohl die Integ-

---

[623] Vgl. Linß /Nutzeffekte/ S. 19-23.

[624] Vgl. Zahavi /Integration/ S. 26-33; IEEE /Glossary/ S. 41.

[625] Vgl. zum folgenden Absatz Linß /Nutzeffekte/ S. 25-27.

[626] Vgl. Kapitel 2.1.

[627] Vgl. zum folgenden Absatz Linß /Nutzeffekte/ S. 23-24.

[628] Vgl. auch Horváth /Controling/ S. 677-679.

rationsreichweite als auch die Integrationsgegenstände und betrachtet, auf welchen Ebenen sich diese befinden und wie viele Ebenen ggf. überwunden werden müssen. Die Integrationsrichtung wird deswegen eine Softwarearchitektur beeinflusst; die Gründe sind dieselben wie bei der Integrationsreichweite (wegen der Regelung der Beziehungen der Softwarekomponenten zueinander) und den Integrationsgegenständen (wegen der Regelung der Gestaltung von Softwarekomponenten).

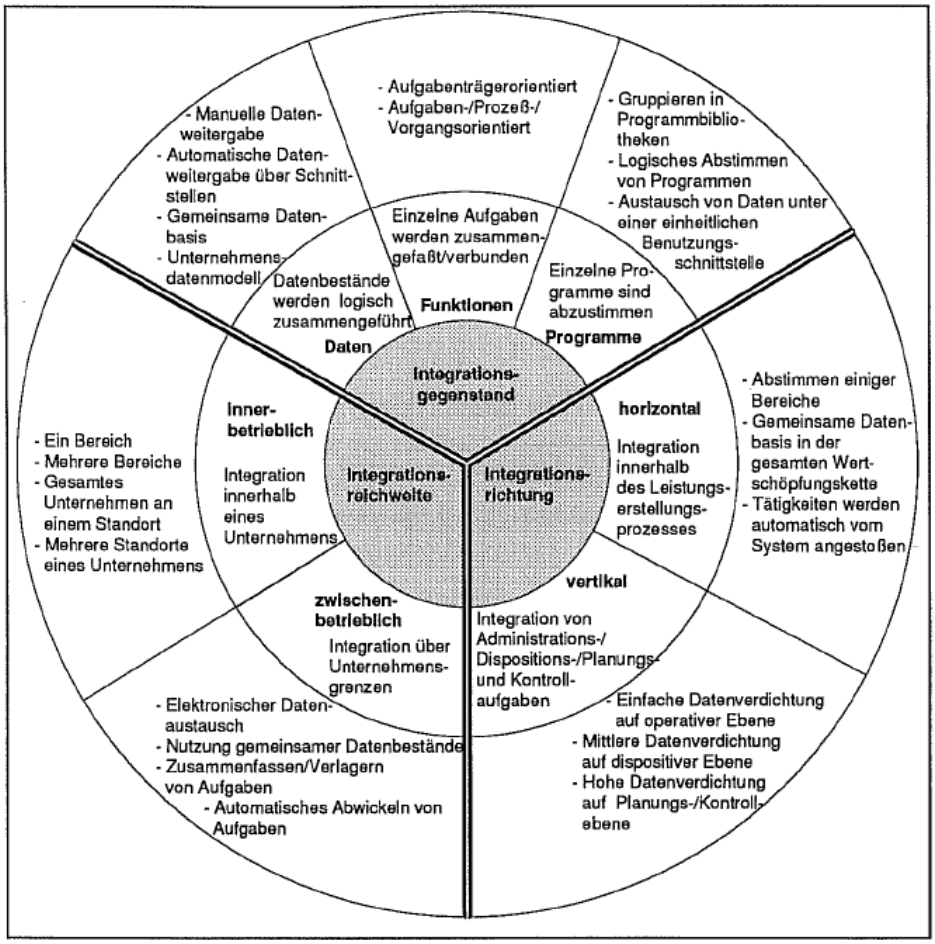

Abb. 3-5: Integrationsbegriff[629]

Die Dimensionen des Bewertungsmodells nach Linß sind teilweise voneinander abhängig. Die Integrationsrichtung ist an die Anzahl potentiell vorhandener Integrationsge-

[629] Linß /Nutzeffekte/ S. 18. (Anm. des Autors: Dies ist eine Darstellung der Graphik, wie sie von Linß verwendet wird. Linß schreibt nach alter Rechtschreibung, wodurch die Graphik ‚Rechtschreibfehler' aufweißt.)

genstände gebunden: z. B. ist eine horizontale Integrationsrichtung nicht möglich, wenn kein Integrationsgegenstand integrationsfähig ist.[630]

Aus diesen Überlegungen leitet sich die Definition des folgenden Kapitels ab.

3.1.4.4.2 Arbeitsdefinition der Integriertheit

Integriertheit einer Softwarearchitektur ist die Fähigkeit, Mechanismen der Konnektivität und Interoperabilität anbieten zu können, damit dem Ökosystem eines Unternehmens Dienstleistungen, Werkzeuge und Technologien mit dem Ziel zur Verfügung gestellt werden können, das Ökosystem zu erhalten, die Leistungsfähigkeit des Ökosystems zu verbessern und an das Unternehmen gestellte Informationsansprüche anforderungsgerecht bedienen zu können.

Unterqualitätsattribute der Integriertheit:

- Integrationsgegenstand
  Zur Bewertung des Integrationsgegenstandes wird die Fähigkeit einer Software untersucht, Daten, Funktionen und Programme verschiedener Unternehmensbereiche und Unternehmensebenen integrieren zu können.
- Integrationsreichweite
  Zur Bewertung der Integrationsreichweite wird die Fähigkeit einer Software untersucht, eine innerbetriebliche und zwischenbetriebliche Integration unter Einbeziehung verschiedener Unternehmensbereiche und -ebenen erreichen zu können.
- Integrationsrichtung
  Zur Bewertung der Integrationsrichtung wird die Fähigkeit einer Software untersucht, horizontale und vertikale Integration zu unterstützen.

3.1.4.4.3 Signifikanz der Integriertheit

Eine Unternehmens-IT kann sowohl bestimmte Funktionsbereiche eines Unternehmens als auch alle betriebswirtschaftlichen Funktionsbereiche und Prozessketten eines Unter-

---

[630] Wobei der Autor dieser Arbeit hier eingesteht, dass dies ein stark polarisierendes Bild der Abhängigkeit der Integrationsdimensionen darstellt. Als Beispiel der Abhängigkeit der Integrationsdimensionen ist dieses Beispiel jedoch hilfreich, da es den Zusammenhang der Integrationsdimensionen klar darstellt.

nehmens unterstützen.[631] Darüber hinaus kann auch die Einbeziehung von Geschäftspartnern in die eigene Unternehmens-IT angestrebt werden.[632] Prozessorientierung, d. h. die Fokussierung auf betriebliche Abläufe (sequentiell abzuarbeitende Funktionen) ggf. über Funktions- und Unternehmensgrenzen hinweg und deren durchgängige Systemunterstützung, ist ein Treiber für die Integration von Unternehmensanwendungen.[633] Gerade die Integration vieler heterogener IT-Systeme ist eines der Hauptziele, das in großen Unternehmen heute angestrebt wird.[634]

Belegt wird die Signifikanz außerdem dadurch, weil moderne Informationssysteme bereits sehr intensiv versuchen, externe Benutzer (z. B. Kunden und Partner) einzubeziehen, sodass es teilweise schwierig wird, externe von internen Nutzern abzugrenzen.[635] Beispielsweise kann bei einem deutschen Versanddienst sowohl der Kunde via Internet, der Bote via mobilem Endgerät, der Angestellte im (nicht zum Unternehmen gehörenden) Ladenlokal als auch der Mitarbeiter an der Telefonhotline einen Versandauftrag inklusive Rechnungsabwicklung erstellen.[636]

Signifikanz der Unterqualitätsattribute

- Integrationsgegenstand

  Integration findet über immer mehr Unternehmensbereiche und zu immer mehr Partnern aus der Unternehmensumwelt statt, die über unterschiedliche Gegenstände zur Integration verfügen. Die Fähigkeit, unterschiedliche Integrationsgegenstände einbeziehen zu können, ist für die Integriertheit einer Unternehmens-IT deswegen von hoher Relevanz.

- Integrationsreichweite

  Die Integrationsreichweite ist durch die immer stärker werdende Vernetzung des Unternehmens mit seiner Umwelt von hoher Bedeutung. Durch eine hohe Integrationsreichweite können heutige und zukünftige Integrationsanforderungen erfüllt werden.

---

631 Vgl. Hansen, Neumann /Wirtschaftsinformatik/ S. 523-524.

632 Vgl. Hansen, Neumann /Wirtschaftsinformatik/ S. 523-524.

633 Vgl. zu diesem Absatz Chappell /Enterprise Service Bus/ 22, 26-27.

634 Vgl. Adam, McKendrick /Everything in Between/ S. 11-12 & S. 21.

635 Vgl. Hansen, Neumann /Wirtschaftsinformatik/ S. 561.

636 Vgl. hierzu die Möglichkeiten der Hermes Logistik Gruppe unter www.hlg.de oder alternativ eine Fallstudie zu Amazon in Hansen, Neumann /Wirtschaftsinformatik/ S. 563-575.

Einstufung der Gewichtung des Qualitätsattributes

Wie aufgezeigt ist die Integriertheit für Unternehmen von großer Relevanz. Aus diesem Grund geht die Bewertung der Integriertheit mit hohem Gewicht in die Bewertung einer Softwarearchitektur ein.

3.1.5 Zusammenfassung des Qualitätsmodells

Zusammenfassend ergibt sich zur Bewertung einer Softwarearchitektur das in Abb. 3-6 dargestellte Qualitätsmodell. In Tab. 3-10 lässt sich erkennen, dass die Mehrzahl der Qualitätsattribute mit einer hohen Gewichtung in die Bewertung dieser Arbeit einfließen wird. Für die Bewertung der Wirtschaftlichkeit einer Softwarearchitektur für einen konkreten Praxisfall kann je nach Anforderungen jedoch ein anderes Gewichtungsbild entstehen.

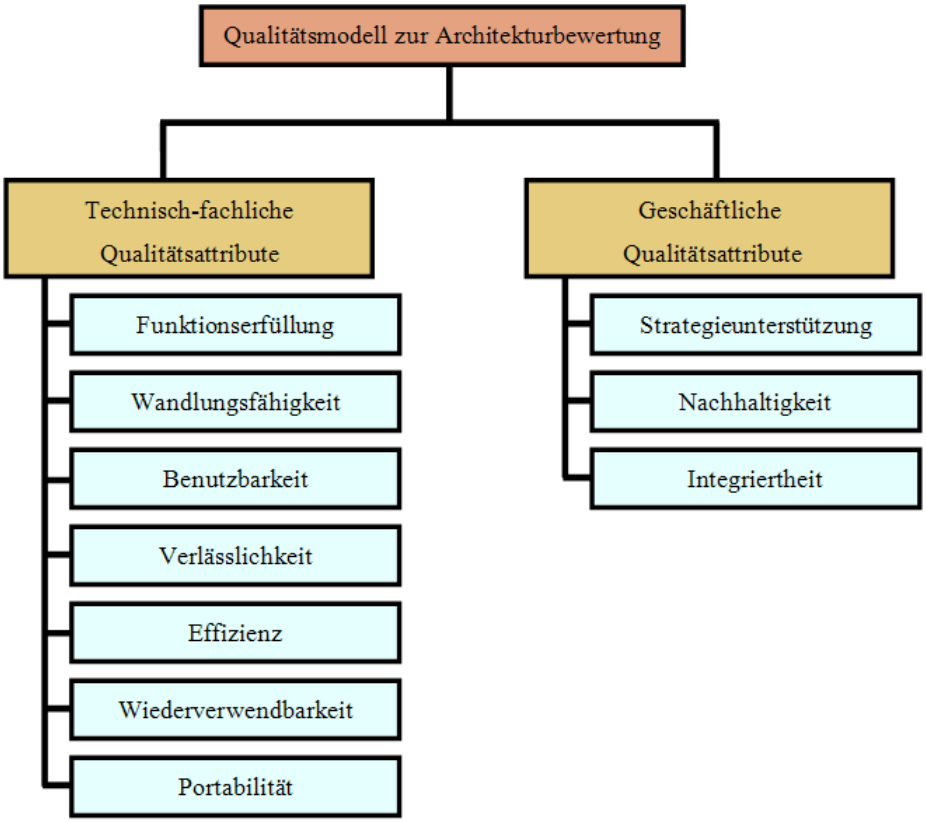

Abb. 3-6: Qualitätsmodell (ohne Unterqualitätsattribute)

Technisch-fachliche Qualitätsattribute		*Gewichtung*[637]
	Funktionserfüllung	v
	Wandlungsfähigkeit	^
	Benutzbarkeit	^
	Verlässlichkeit	^
	Effizienz	v
	Wiederverwendbarkeit	^
	Portabilität	^
Geschäftliche Qualitätsattribute		***Gewichtung***
	Strategieunterstützung	^
	Nachhaltigkeit	^
	Integriertheit	^

Tab. 3-10: Gewichtung der Qualitätsattribute im Qualitätsmodell

3.2 Bewertung des Nutzenpotentials einer SOA

In einer Umfrage zum Nutzenpotential einer SOA mit fast 1000 Managern und Entwicklern berichteten viele der Befragten, dass es abzusehen sei, Nutzen in Form von Wiederverwendung, Entwicklungsproduktivität und Kostenersparnissen realisieren zu können.[638] Um diese Einschätzung von Experten zu verstehen, wird in diesem Kapitel aus theoretischer Sichtweise und anhand des in Kapitel 3.1 aufgestellten Bewertungsmodells das Nutzenpotential einer SOA untersucht.

SOA stellt entsprechend dem Verständnis dieser Arbeit[639] eine unternehmensweite Softwarearchitektur dar. Dementsprechend lassen sich im Umfeld der Softwarearchitektur das Geschäft, die IT, die Mitarbeiter und die dazwischen bestehenden (Integrations-

[637] Legende: v niedriges Gewicht, ^ hohes Gewicht, **0** neutrales Gewicht.

[638] Vgl. Adam, McKendrick /Everything in Between/ S. 3.

[639] Vgl. Kapitel 2.1.

)Aufgaben, wie z. B. die Business-IT-Integration, als Betroffene identifizieren.[640] In Bezug auf dieses Umfeld können Nutzen- und Kostenpotentiale aufgedeckt werden. Aus diesem Grund lässt sich eine unternehmensweite Systemintegration nicht auf die Betrachtung der technischen Komponenten beschränken.[641] Es ist ersichtlich, dass Nutzenpotentiale sowohl in strategische als auch in operative Potentiale unterschieden werden können.[642] In der Literatur zur Nutzenbewertung einer Unternehmens-IT wird ebenfalls diskutiert, dass der Nutzen nicht nur anhand der Produktivität einer Unternehmens-IT bemessen werden kann.[643] Es kommt vor, dass sich ein erhöhter Nutzen herausbildet, wenn durch eine integrierte Gestaltung von Geschäftsprozessen gegenüber Wettbewerbern ein Wettbewerbsvorteil entsteht, selbst wenn die Produktivität der Unternehmens-IT dadurch nicht erhöht wird. Auch hier wird deutlich, dass strategische Nutzenpotentiale einer Unternehmens-IT vorliegen und bewertet werden müssen.

Wie schon dargestellt entspricht dieser Sichtweise auch das in Kapitel 3.1 aufgestellte Qualitätsmodell. Die Bewertung der Nutzenpotentiale einer SOA erfolgt unter Verwendung des in Kapitel 3.1 aufgestellten Qualitätsmodells für Softwarearchitekturen. Software, die jeweils auf Basis einer bestimmten Softwarearchitektur aufgebaut ist, kann sich dennoch in Einzelheiten unterscheiden und andere Schwerpunkte setzen, die in der Realisierung beachtet werden. Die Untersuchungen von Permutationen möglicher Schwerpunkte bezüglich der Verfolgung von Gestaltungszielen und Priorisierung von Gestaltungsvorgaben einer SOA würde zum einen zu unterschiedlichen Bewertungen führen, und zum anderen würde eine hohe Anzahl zu bewertender Gestaltungsalternativen entstehen. Infolgedessen wird eine Softwarearchitektur nach dem ‚Idealtyp' einer SOA bewertet. Der Idealtyp folgt allen im Kapitel 2 vorgestellten Gestaltungsrichtlinien und kann alle Aspekte der Gestaltungsrichtlinien, wie z. B. synchrone und asynchrone Kommunikation, unterstützen.

Im Folgenden werden in Kapitel 3.2.1 zunächst Bewertungsalternativen und –probleme diskutiert. Anschließend werden Nutzenpotentiale entsprechend dem Qualitätsmodell aus Kapitel 3.1 im Kapitel 3.2.2 technisch-fachlichen und in Kapitel 3.2.3 geschäftlichen Nutzenpotentialen identifiziert. Kapitel 3.2.4 normalisiert die aufgestellten Bewer-

---

[640] Vgl. Klement /Enterprise Architecture/ S. 8-14.

[641] Aier, Schönherr /Flexibilisierung/ S. 5.

[642] Vgl. Aier, Schönherr /Flexibilisierung/ S. 14.

tungen. Kapitel 3.2.5 stellt dar, welche Nutzenpotentiale aus der Verwendung von Standards resultieren können.

3.2.1 Problematik der Bewertung von Softwarearchitekturen

Nutzen kann unterschieden werden nach dem Entstehungsort in direkten und indirekten Nutzen sowie nach monetärer Bewertbarkeit in direkt, schwer und nicht quantifizierbaren Nutzen.[644] Im Allgemeinen und insbesondere bei komplexen IT-Projekten stößt die Bewertung der Wirtschaftlichkeit von Informationssystemen auf beträchtliche Probleme: Ungewissheit der künftigen Kosten und des zu erwartenden Nutzens, Quantifizierungsproblem insbesondere des Nutzens, Komplexitätsproblem insbesondere durch Interdependenzen, Systemabgrenzungsproblemen und Zeitrestriktionsproblem. Die Problematik der Kostenbewertung wird im Kapitel 4.1 behandelt. Die verbleibenden Problematiken werden im Folgenden in zwei Bereiche differenziert: zum einen in Probleme durch den zu bewertenden Gegenstand, in diesem Fall der ‚Softwarearchitektur' (Kapitel 3.2.1.1), und zum anderen in Probleme der Quantifizierung (Kapitel 3.2.1.2).

3.2.1.1 Problematik des Bewertungsgegenstands ‚Softwarearchitektur'

Wie schon festgestellt, wurde in Studien der vergangenen Jahrzehnte wiederholt versucht, wirtschaftlichen Nutzen der IT auf unterschiedlichen Betrachtungsebenen nachzuweisen. Beispielsweise beschrieben Sassone und Schaffer vor zwei Jahrzehnten die Schwierigkeiten der Bewertung von Investitionen in Desktopcomputer.[645] Diesen Studien folgend konnte eine Produktivitätssteigerung durch den Einsatz von IT in den 80er und 90er Jahren nicht nachgewiesen werden, vielmehr wurde damals unter dem Begriff „Productivity Paradox"[646] (Produktivitätsparadox) festgehalten, dass trotz hoher Investitionen in IT keine Produktivitätssteigerung stattfindet.[647] Diese Studien erzielten jedoch teilweise gegensätzliche und häufig nur unter Annahmen zutreffende Ergebnisse. Daraus wurde der Schluss gezogen, dass es nicht zweifelsfrei möglich ist, wirtschaftlichen

---

[643] Vgl. zum folgenden Absatz Brynjolfsson, Yang /Productivity/ S. 208.

[644] Vgl. zum folgenden Absatz Horváth /Controlling/ S. 712-715; Antweiler /Wirtschaftlichkeitsanalyse/ S. 2-3; Nagel /Nutzen/ S. 26; Horváth /Grundprobleme/ S. 2-3.

[645] Vgl. zu diesem Absatz Sassone /CBA of IS/ S. 126 & S. 131.

[646] Vgl. Brynjolfsson /Productivity Paradox/ S. 67-77; Brynjolfsson, Hitt /Paradox Lost/ S. 41.

[647] Vgl. Brynjolfsson, Yang /Productivity/ S. 179-180; Lewin, Hunter /Information Technology/ S. 274.

Nutzen durch Investition in IT statistisch signifikant nachzuweisen.[648] Erklärungen für das Produktivitätsparadox lassen sich hauptsächlich auf folgende Gründe zurückführen:[649]

- Messprobleme
- zeitliches Auseinanderfallen von Investition und Nutzeffekten
- unzureichende Kenntnis der Auswirkungen des IT-Einsatzes
- IT ist nur ein Einflussfaktor auf den Unternehmenserfolg sowie
- unzureichende Abstimmung zwischen IT und Organisation.[650]

Neuere Studien zeigen jedoch auf, dass sich ein positiver Effekt von Investitionen in IT auf den Unternehmenserfolg feststellen lässt,[651] obwohl ebenfalls festgehalten wird, dass es weiterhin schwierig ist, wirtschaftlichen Nutzen der IT nachzuweisen, hauptsächlich, weil weitreichende Investitionen in IT getätigt werden.[652]

Die Problematik der Bewertung einer Softwarearchitektur lässt sich auf die Eigenschaft einer Softwarearchitektur zurückführen, die Grundlage einer Unternehmens-IT darzustellen und Software-Entwicklern eine Anleitung zur Softwareentwicklung zu geben. Beides hat Auswirkungen auf die Qualität einer Unternehmens-IT und auf den Nutzen einer Unternehmens-IT.[653] Erklärungen für Bewertungsprobleme liefern sowohl vorhandene Investitionsbewertungsmethoden und lassen sich auch aus Bewertungsproblemen einer Unternehmens-IT ableiten.

Probleme der Bewertung von Softwarearchitekturen existieren, weil verfügbare Investitionsbewertungsmethoden die Komplexität der Investition in Softwarearchitekturen

---

[648] Vgl. Alshawi, Irani, Baldwin /Benchmarking/ S. 414-415; Smithson, Hirschheim /Old Problem/ S. 160-165; Irani, Love /Frame of Reference/ S. 74-75; Potthof /Empirische Studien/ S. 58-63; Shin /Strategic Choice/ S. 227-229.

[649] Vgl. Potthof /Empirische Studien/ S. 58-63; Linß /Nutzeneffekte/ S. 41-42.

[650] Eine weitere Hypothese, für deren Aussage keine Überprüfung in der Literatur gefunden werden konnte, ist, dass Studien, die die Performanz ganzer Wirtschaftszweige anhand der Investitionen in IT beurteilt, nicht berücksichtigen (können), welche Performanz sich ohne Einsatz von IT ergeben hätte. Weil Unternehmen der einzelnen Wirtschaftsbereiche – der Argumentation von Carr (vgl. Carr /IT/) folgend – IT als Alltagsgegenstand einsetzen (müssen), um am Markt bestehen zu können, ist eine solche Untersuchung vermutlich nicht möglich.

[651] Vgl. Potthof /Empirische Studien/ S. 57-58; Brynjolfsson, Hitt /Paradox Lost/ S. 61-63; Lucas /Information Technology/ S. 43-78 & S. 81-94.

[652] Vgl. Irani, Love /Frame of Reference/ S. 74.

nicht beachten.[654] Diese Komplexität ist zurückzuführen auf die Gegebenheit, dass eine Softwarearchitektur die gesamte Unternehmens-IT und alle erstellten Softwarekomponenten beeinflusst. Weiterhin erhöhen zeitliche Aspekte die Komplexität: Weil eine Unternehmens-IT lange in Betrieb ist muss ein weiter Horizont zur Bewertung angesetzt werden. Folglich sind auch zukünftige Entwicklungen von Softwarekomponenten einer Unternehmens-IT betroffen. Dadurch kommt es zu vielfältigen Auswirkungen auf und zwischen Softwarekomponenten einer Unternehmens-IT, die in Investitionsmethoden nicht ausreichend beachtet werden.

Des Weiteren unterliegt die Bewertung des Nutzens von Softwarearchitekturen mindestens den gleichen Problemen wie die Bewertung des Nutzens einer Unternehmens-IT. Es wird nämlich noch schwieriger, einzelne Investitionen in eine Unternehmens-IT, wie eben in eine Softwarearchitektur, dem Gesamterfolg eines Unternehmens zuzurechnen als vergleichsweise die Gesamtinvestition in eine Unternehmens-IT. Weil grundlegende Gestaltungsrichtlinien inklusive entsprechender Software zur Unterstützung der Gestaltungsrichtlinien[655] eingeführt werden und sich diese langfristig auswirken, unterliegt die Bewertung der gleichen Zeitproblematik wie der von Investitionen in eine Unternehmens-IT.

Farbey, Land und Targett haben die Komplexität unterschiedlicher Investitionen in IT untersucht und eine Kategorisierung des Problems zur Nutzenbewertung aufgestellt.[656] Ihr Forschungsergebnis spiegelt die oben genannten Probleme anschaulich in einer sog. „Benefits Evaluation Ladder"[657] (siehe Abb. 3-7) wider. Die „Benefits Evaluation Ladder" unterteilt Investitionen in eine Unternehmens-IT danach, wie sicher diese Investitionen bewertet werden können. Sie kommen jedoch auch zu dem Ergebnis, dass die Ebenen der „Benefits Evaluation Ladder" zusätzlich darstellen, dass eine höhere Wahrscheinlichkeit besteht, größeren Nutzen auf den oberen Stufen zu erreichen als auf niedrigeren Stufen. Gleichzeitig stellen sie fest, dass potentiell höhere Risiken und hö-

---

653 Vgl. Bass, Clements, Kazman /Software Architecture/ S. 29-30.

654 Vgl. zu diesem Absatz Irani, Love /Frame of Reference/ S. 76-78.

655 In einer SOA muss z. B. Software zur Realisierung der Kommunikationsinfrastruktur eingeführt werden. Zur Kommunikationsinfrastruktur siehe Kapitel 2.2.4.3.

656 Weitere Einteilungen können z. B. bei Hochstrasser /IT investments/ S. 216-219 oder Fitzgerald /Evaluating/ S. 16-19 gefunden werden. Die „Benefits Evaluation Ladder" nach Farbey, Land und Targett wird an dieser Stelle wegen des Einbezugs einer gesamten Unternehmens-IT verwendet.

657 Vgl. zu diesem Absatz Farbey, Land, Targett /Taxonomy/ S. 42.

here Schwierigkeiten der Durchsetzung entsprechender Investitionen gegenüber Entscheidungsträgern auf den höheren Stufen auftreten können. Auf der ersten Stufe („Mandatory Changes") werden unter anderem zwingend notwendige Änderungen eingestuft.[658] Diese Änderungen sind z. B. solche, die aus legalen Gründen (z. B. Gesetzesänderungen, Eintritt in neue Märkte mit unterschiedlicher Gesetzeslage) umgesetzt werden müssen oder um im Wettbewerb bestehen zu können. Eine Bewertung kann hier oftmals leicht erfolgen. Typische Fragestellungen zur Bewertung betreffen auf dieser Ebene die Wahl der einzusetzenden Technologien oder Prozesse.

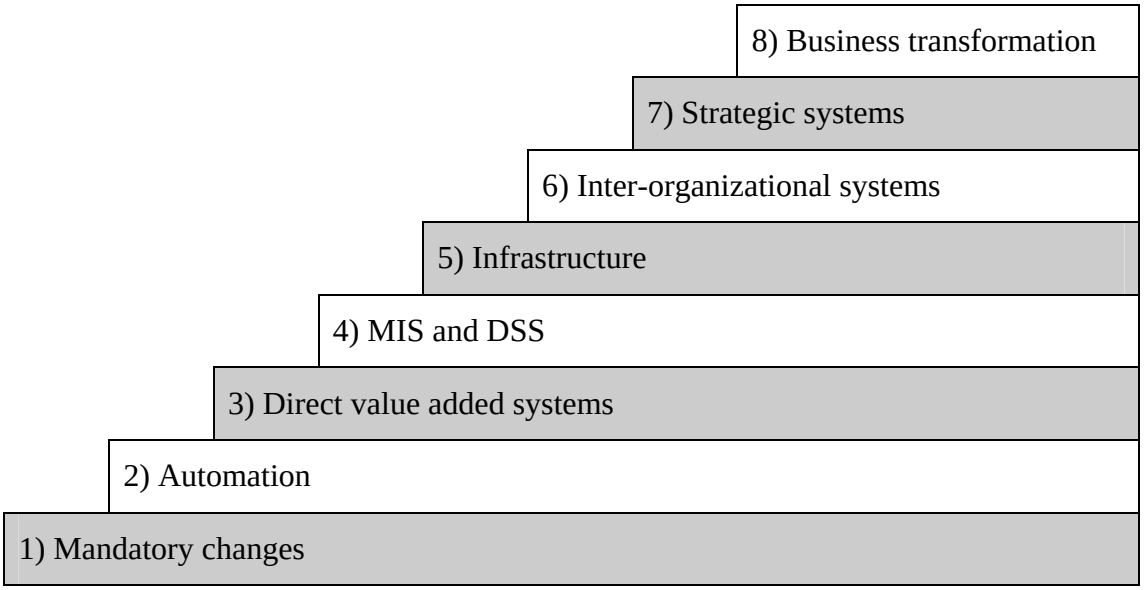

Abb. 3-7: „Benefits Evaluation Ladder"[659]

Zur Erläuterung, warum die Bewertung einer SOA Probleme aufwirft, kann nicht nur auf eine Stufe der „Benefits Evaluation Ladder" zurückgegriffen werden. Wie im Folgenden dargestellt betrifft die Bewertung einer SOA die vier oberen, nach Farbey, Land und Targett also die vier am schwierigsten zu bewertenden Stufen.

Im Folgenden wird dargestellt, wie die vier oberen Stufen der „Benefits Evaluation Ladder" definiert sind, warum die Bewertung einer SOA diese Stufen betrifft und welche Auswirkungen sich dadurch auf die Bewertung ergeben.[660] Diese vier Stufen der

---

[658] Vgl. zur ersten Stufe Farbey, Land, Targett /Taxonomy/ S. 42-43.

[659] Farbey, Land, Targett /Taxonomy/ S. 42.

[660] Der interessierte Leser möge auf die Arbeit von Farbey, Land und Targett zurückgreifen, um die Darstellung der kompletten Einstufung nachzulesen. Vgl. Farbey, Land, Targett /Taxonomy/.

„Benefits Evaluation Ladder" sind die Stufen ‚Infrastruktur' (Infrastructure), ‚zwischenorganisatorische Systeme' (Inter-organizational systems), ‚strategische Systeme' (Strategic systems) und ‚Geschäftstransformation' (Business transformation):

- 5. Stufe: Infrastruktur (Infrastructure)[661]

  Auf der Ebene der Infrastruktur werden Investitionen zusammengefasst, die einer Unternehmens-IT grundlegende Fähigkeiten zur Verfügung stellt und i. d. R. keine anwendungsspezifischen funktionalen Anforderungen erfüllen. Solche Investitionen stellen Funktionen für eine Unternehmens-IT zur Verfügung, die z. B. die Fähigkeit zur Kommunikation oder zur sicheren Aufbewahrung von Daten bereitstellt. Solche Investitionen lassen sich zum Zeitpunkt der Investition nur schwer bewerten, weil sie öfters keinen direkten Nutzen stiften. Es kann sogar davon ausgegangen werden, dass unklar ist, welche Auswirkungen eine solche Einführung überhaupt haben wird.

  Eine Investition in Softwarearchitekturen kann als Infrastrukturmaßnahme aufgefasst werden, weil über das Festlegen der Gestaltung der Beziehungen von Softwarekomponenten untereinander und durch die Art der Aufteilung von Funktionen auf Softwarekomponenten Maßnahmen der IT-Infrastruktur betroffen sind. Hier müssen z. B. passende Kommunikationsmöglichkeiten geschaffen und die Maßnahmen getroffen werden, die die Funktionsfähigkeit von Softwarekomponenten (z. B. eine Laufzeitumgebung) ermöglichen.

- 6. Stufe: Zwischenorganisatorische Systeme (Inter-organizational systems)[662]

  Zwischenorganisatorische Systeme überbrücken die Grenzen eines Unternehmens zu anderen Unternehmen und verbinden mehrere Unternehmen über ihre Unternehmens-IT miteinander. Nutzen soll durch eine verbesserte Zusammenarbeit, z. B. durch schnelleren Datenaustausch, erreicht werden. Allerdings haben zwischenorganisatorische Systeme vielfältige Auswirkungen auf die teilnehmenden Partner. Dies betrifft z. B. gegenseitige Abhängigkeiten, weil durch eine Änderung des Systems immer alle Partner betroffen sind. Ferner können Probleme der Abstimmung durchzuführender Änderungen auftreten. Auch können stärkere Partner Druck ausüben, um Änderungen durchzusetzen. Das einzelne Unternehmen verliert so an Gestaltungs- und Handlungsfreiräumen. Eine Nutzenabwägung zwischen höherer Gestal-

---

[661] Vgl. zu diesem Absatz Farbey, Land, Targett /Taxonomy/ S. 46-47.

[662] Vgl. zu diesem Absatz Farbey, Land, Targett /Taxonomy/ S. 47.

tungs- und Handlungsfreiheit und potentiellem Nutzen durch gemeinsam genutzte zwischenbetriebliche Systeme ist nur schwer möglich; und zwar wegen der Beteiligung externer Unternehmen, die wiederum an der eigenen Gewinnmaximierung und der Vertretung der eigenen Interessen interessiert sind.

Softwarearchitekturen haben großen Einfluss auf die Fähigkeiten und Möglichkeiten, zwischenorganisatorische Systeme aufzubauen. Durch Gestaltungsrichtlinien, wie Softwarekomponenten miteinander kommunizieren, und durch den Einfluss von Softwarearchitekturen auf die Infrastruktur einer Unternehmens-IT werden die Möglichkeiten zur Interaktion von der eigenen zu einer anderen Unternehmens-IT beeinflusst. Unternehmen arbeiten mittlerweile auf IT-Ebene zusammen,[663] und setzen Softwarelösungen ein, um eine solche IT-Kooperation im operativen Geschäft betreiben zu können. Die Möglichkeiten der Einbindung von Systemen zur Unterstützung solcher IT-gestützer zwischenbetrieblicher Kooperation wird durch Softwarearchitekturen – insbesondere durch den Einfluss auf die Infrastruktur einer Unternehmens-IT – ebenfalls eingeschränkt.

- 7. Stufe: Strategische Systeme (Strategic systems)[664]

  Strategische Systeme sollen Wettbewerbsvorteile sichern, z. B. durch erhöhte Produktivität, durch die Einführung neuer Management- und Organisationsmethoden und durch die Entwicklung neuer Geschäftsfelder und -abläufe. Die Realisierung solcher Wettbewerbsvorteile erfordert innovative Vorgehensweisen und vor allem auch die Erkenntnis des Managements, dass eine Unternehmens-IT als strategische Ressource verwendet werden kann. Eine Bewertung strategischer Systeme ist extrem schwierig, weil strategische Vorgehensweisen mit der Hoffnung verbunden sind, entsprechend geplante Auswirkungen zu zeigen und entsprechend erwartete Wettbewerbsvorteile tatsächlich zu realisieren.[665]

  Der Einfluss von Softwarearchitekturen auf strategische Systeme ist von indirekter Natur. Strategische Systeme werden auf lange Frist geplant, jedoch kann die Notwendigkeit bestehen, solche Systeme sehr kurzfristig umsetzten zu müssen. Beispielsweise wird die strategische Entscheidung einen neuen Vertriebsweg in dem Unternehmen zu implementieren langfristig ausgelegt sein. Die Notwendigkeit einer

---

[663] Vgl. hierzu beispielsweise den Einsatz von EDI.

[664] Vgl. zu diesem Absatz Farbey, Land, Targett /Taxonomy/ 47-48.

kurzfristigen Umsetzung jedoch kann, z. B. aus Gründen des Wettbewerbs, gegeben sein. Eine Softwarearchitektur hat Einfluss auf strategische Systeme, weil sie es ermöglichen muss, sowohl langfristig Software unterstützen als auch kurzfristig Software einführen und bereits eingesetzte Software (unter Umständen auch kurzfristig) ändern zu können.

- 8. Stufe: Geschäftstransformation (Business transformation)[666]
  Geschäftstransformation betrifft das gesamte Unternehmen. Häufig werden im Zuge von Geschäftstransformationen fundamentale Änderungen an der Struktur und den Prozessen eines Unternehmens durchgeführt. Notwendig werden solche Änderungen z. B. durch eine starke Änderung des unternehmerischen Umfeldes oder durch stärker werdenden Wettbewerb. Ebenfalls ist die Unternehmens-IT von Änderungen aufgrund von Geschäftstransformationen betroffen. Solche Änderungen können direkte Änderungsanforderungen durch neue funktionale und nicht-funktionale Anforderungen sowie andere indirekte Änderungsanforderungen durch geänderte zu unterstützende Organisationsstrukturen und Geschäftsprozesse darstellen.
  Analog zur Argumentation der 7. Stufe kann auch hier ein indirekter Einfluss einer Softwarearchitektur auf die Fähigkeit zur Geschäftstransformation aufgestellt werden: Erstens wird ebenso ein langfristiger Zeithorizont betrachtet und eine Geschäftstransformation muss ggf. schnell erfolgen. Zweitens werden Geschäftsprozesse heute zum großen Teil durch Software unterstützt, und Geschäftstransformationen beinhalten die Änderung und Neustrukturierung von Geschäftsprozessen und folglich auch die Änderung und Neustrukturierung der entsprechenden Unternehmens-IT.

Wie gezeigt werden konnte, ist die Bewertung des Nutzenpotentials einer Softwarearchitektur durch die vielfältigen Auswirkungen des Bewertungsgegenstandes ‚Softwarearchitektur' problematisch.

3.2.1.2 Problematik der Bewertungsskala (Quantifizierung)

Der Begriff Nutzen bezeichnet allgemein die Wirkungen, die einem Unternehmen im Hinblick auf seine Zielerreichung durch eine Maßnahme entstehen.[667] Diese Wirkungen

---

[665] Vgl. beispielsweise Porter /Strategy/ S. 324-325.

[666] Vgl. zu diesem Absatz Farbey, Land, Targett /Taxonomy/ 48-49.

[667] Vgl. Linß /Nutzeffekte/ S. 30-31.

können Gegenstände und Prozesse beeinflussen und auch negativ sein, sind jedoch nicht notwendigerweise mit Kosten[668] gleichzusetzen. Nutzenpotentiale sind somit die potentiell auftretenden Wirkungen, die zur Zielerreichung durch eine Maßnahme angestrebt werden. Ein Nutzenpotential kann positiv, neutral als auch negativ ausgeprägt sein und wird beschrieben durch eine Bezeichnung und eine entsprechende Bewertung, die die Stärke angibt, mit der der beschriebene Effekt zu erwarten ist.[669] Im Folgenden wird nicht mehr explizit zwischen Nutzen und Nutzenpotential unterschieden und stattdessen angenommen, dass der Nutzen genau wie das Nutzenpotential auch eine Erwartungshaltung darstellt.

Nutzenpotential kann unterschieden werden nach dem Entstehungsort in direkte und indirekte Nutzenpotentiale sowie nach monetärer Bewertbarkeit in direkt, schwer und nicht quantifizierbare Nutzenpotentiale.[670] Direkte Nutzenpotentiale sind Auswirkungen, die unmittelbar einem Nutzen zugeordnet werden können. Indirekte Nutzenpotentiale können nur mittelbar zugeordnet werden. Z. B. kann eine Erhöhung der Mitarbeiterzufriedenheit noch unmittelbar erkannt werden, wogegen eine Produktivitätserhöhung durch zufriedene Mitarbeiter nur noch mittelbar aufgestellt werden kann. Monetär schwer bzw. nicht quantifizierbare Nutzenpotentiale werden i. d. R. als intangible Werte bezeichnet.[671] Eine Unternehmens-IT weist i. d. R. intangible Werte auf.[672] Selbst eine Bewertung direkt quantifizierbarer Nutzenpotentiale kann sich als ungenau erweisen, weil intangible Nutzenpotentiale einen Einfluss auf quantifizierbare haben können. [673]

Die Bewertung von Nutzenpotentialen einer Softwarearchitektur (somit auch einer SOA) als monetär bewertbare Nutzenpotentiale ist schwierig, weil viele Nutzenpotentiale monetär schwer oder nicht quantifizierbar sind. Zachman stellt deswegen die Behauptung auf, dass Softwarearchitektur nicht über Kosten gerechtfertigt werden kann[674] und

---

668 Die Kosten, die im Zusammenhang einer SOA anfallen, werden in Kapitel 1 untersucht. Kosten einer Unternehmens-IT fallen z. B. bei Entwicklung, Einführung, Betrieb und Erweiterung an. Vgl. hierzu z. B. Sommerville /Software Engineering/ S. 24-26; Schumann /Nutzeffekte/ S. 59.

669 Vgl. Linß /Nutzeffekte/ S. 30-31.

670 Vgl. Horváth /Controlling/ S. 712-715.

671 Vgl. Boer /Valuation/ S. 72; Reifer /Business Case/ S. 61; Hares, Royle /Measuring/ S. 5; Irani, Love /Frame of Reference/ S. 77-78; Boehm /Economics/ S. 223; Murphy, Simon /Intangible Benefits/ S. 305-308.

672 Vgl. Lucas /Information Technology/ S. 9, S. 13-17 & S. 81-94.

673 Vgl. Heimann, Kappes /Kostenfalle/ S. 47; Alshawi, Baldwin, Irani /Benchmarking/ S. 419.

674 Vgl. Zachman /Cost-Justify Architecture/ S. 2.

die META-Group stellt fest, dass über den Einsatz einer SOA nicht nur auf Basis von Kostenersparnissen entschieden werden kann.[675] Irani und Love stellen explizit heraus, dass eine Investition in eine SOA eine strategisch relevante, und somit monetär schwer bewertbare Investition ist, die für das entsprechende Unternehmen Auswirkungen auf das zukünftige Geschäft und die zukünftige Wettbewerbslage haben wird.[676]

Die Vorgehensweise zur Bewertung der Ausprägung eines Nutzenpotentials dieser Arbeit baut auf Argumentationsbilanzen und Nutzeffektketten auf. Generell geeignet zur Bewertung qualitativer Ausprägungen sind Nutzwertanalysen, Argumentationsbilanzen und Nutzeffektketten.[677] Nutzwertanalysen, wie z. B. die Szenariotechnik, benötigen zur Bildung von Szenarien eine konkrete zu bewertende Situation und sind deswegen für diese Arbeit nicht geeignet, weil die Wirtschaftlichkeit einer SOA unter einer allgemeinen Sicht analysiert werden soll.[678] Darüber hinaus sind auch die Methoden der Nutzwertanalysen wie z. B. die Entwicklungen des SEI ‚ATAM' und ‚CBAM' nicht ausgereift. Deutlich wird dies dadurch, weil diese beiden Methoden vielfältig überarbeitet wurden und werden, sobald sie zum Einsatz kommen.[679] Dies ist ein Hinweis darauf, dass eine einheitliche und anerkannte Bewertungsmethode nicht einfach gefunden werden kann – und aktuell noch nicht gefunden worden ist.

Die Problematik der Bewertungsskala ist für Softwarearchitekturen durchaus gegeben, weil Nutzenpotential einer Softwarearchitektur existiert und durchaus unterschiedlich ausgeprägt sein kann. Bass, Clements und Kazman stellen z. B. fest, dass Softwarearchitekturen generell Einfluss auf den Erfolg von Software und auf den Erfolg des unterstützten Geschäfts haben.[680] Reinertsen und Smith verallgemeinern dies noch weiter,

---

675 Vgl. Sholler /Case/ S. 2; Beispielsweise Reifer, Hares und Royle sowie Sassone und Schaffer bewerten Softwarearchitekturen dementsprechend und beachten monetär schwer oder nicht bewertbare Nutzenpotentiale. Vgl. Reifer /Business Case/ S. 61; Hares, Royle /Measuring/ S. 5. Sassone, Schaffer /CBA/ S. 32-40.

676 Vgl. Irani, Love /Frame of Reference/ S. 77.

677 Vgl. zur Eignung von Verfahren zur Nutzenbewertung Linß /Nutzeffekte/ S. 51-59; Horváth /Controlling/ S. 713-715; Schumann /Wirtschaftlichkeitsbeurteilung/ S. 167-178.

678 Vgl. zum Einsatz der Szenariotechnik zur Architekturbewertung Abowd u. a. /Scenario-based Analysis/ S. 48-49, Asundi, Kazman, Klein /Economic Approach/ S. 23; Kazman, Bass /Architecture Reviews/ S. 70-71.

679 Vgl. beispielsweise Asundi, Kazman, Klein /Quantifying/ S. 299; Asundi, Kazman, Klein /Economic Considerations/ S. 5-6; Asundi, Kazman, Klein /Economic Approach/ S. 5-8; Clements, Kazman, Klein /ATAM/ S. 7-8; Bass, Clements, Kazman /Software Architecture/ S. 271-288 & S. 307-316.

680 Vgl. Bass, Clements, Kazman /Software Architecture/ S. 29-30.

indem Sie darstellen, dass die jeweils zugrunde gelegte Architektur für jedes Produkt (also z. B. nicht nur für Software im Falle einer Softwarearchitektur, sondern im Falle der Bauarchitektur auch für Häuser) Einfluss auf den Nutzen des entsprechenden Produktes hat.[681]

3.2.1.3 Bewertungsvorgehen

Wie zu Beginn von Kapitel 3.2 dargestellt ist die Voraussetzung für die Nutzenbetrachtung dieser Arbeit, dass die in Kapitel 2 vorgestellten Gestaltungsprinzipien und –ziele (der ‚Idealtypus' einer SOA) verfolgt werden. Allerdings kann nicht davon ausgegangen werden, dass im Praxiseinsatz alle Gestaltungsprinzipien und –ziele verfolgt werden. Aus diesem Grund werden bei der Bewertung alle Gestaltungsprinzipien und –ziele kritisch hinterfragt und es wird dargestellt, welches Nutzenpotential sich ergibt, wenn nicht alles verfolgt wird.

Um die Problematik der Bewertungsskala zu vereinfachen, werden die Ausprägungen der einzelnen Nutzenpotentiale im Folgenden in einer dreistufigen Ordinalskala bewertet. Die Verwendung grober Bewertungspunkte ermöglicht Aussagen auch in ansonsten kritischen Bewertungsfällen. Dies ist für diese Arbeit notwendig, weil auf allgemeiner Ebene von Besonderheiten einzelner Unternehmen und einzelner Anwendungsfälle abstrahiert wird, obwohl bestimmte Leistungsmerkmale der Unternehmens-IT für einzelne Unternehmen unterschiedlich wichtig sind.[682] Dieses Vorgehen zur Bewältigung der Problematik der Bewertungsskala bringt den Nachteil mit sich, dass erstens die Bewertung nicht für alle Unternehmen zutreffend sein wird; zweitens ermöglicht eine Ordinalskala nur qualitative Aussagen über Wirkungen, und es kann lediglich festgestellt werden, ob und in welche Richtung Nutzenpotentiale wirken.

Im Folgenden wird das Nutzenpotential einer SOA anhand des in Kapitel 3.1 vorgestellten Qualitätsmodells aus technisch-fachlicher sowie geschäftlicher Sicht untersucht.

---

[681] Vgl. Reinertsen, Smith /Half the Time/ S. 116.

[682] Teubner, Rentmeister und Klein argumentieren analog bei der Erstellung eines Evaluationsinstrumentariums für Informationssysteme, vgl. Teubner, Rentmeister, Klein /IT-Fitness/ S. 77.

3.2.2 Nutzenpotentiale aus technisch-fachlicher Sicht

Im Folgenden wird die SOA anhand der technisch-fachlichen Qualitätsattribute[683] bewertet.

3.2.2.1 Funktionserfüllung

Die Funktionserfüllung ist die Fähigkeit eines Softwareproduktes, die Funktion zur Verfügung zu stellen, welche explizit erwähnte und implizit erwartete Anforderungen erfüllt, wenn die Software unter festgelegten Bedingungen genutzt wird.[684]

Im Folgenden werden auf Basis von Argumentationen Bewertungen der Unterqualitätsattribute der Funktionserfüllung vorgenommen. Anschließend werden die Wirkungen einer SOA auf die Funktionserfüllung einer Unternehmens-IT untersucht und es wird eine Gesamtbewertung aufgestellt.

3.2.2.1.1 Nutzen der Unterqualitätsattribute der Funktionserfüllung

Die Unterqualitätsattribute der Funktionserfüllung sind Sicherheit, Interoperabilität und Gehorsamkeit.[685]

Sicherheit

Sicherheit ist die Fähigkeit einer Software, Informationen und Daten so zu schützen, dass nicht autorisierte Personen oder Systeme diese weder lesen noch ändern können, wohingegen autorisierten Personen oder Systemen der Zugang erlaubt wird.[686]

Die Sicherheit einer SOA wird negativ beurteilt. Eine SOA ermöglicht den Aufbau einer verteilten Unternehmens-IT.[687] Eine solche Verteilung kann durchaus über verschiedene Standorte hinweg bis hin zu Marktpartnern erfolgen. In diesem Fall ist damit zu rechnen, dass Kommunikation der Unternehmens-IT über Netze geführt wird, welche nicht unter 100%iger Kontrolle des einsetzenden Unternehmens stehen. Die Anforde-

---

[683] Vgl. zur Darstellung technisch fachlicher Qualitätsattribute Kapitel 3.1.3.

[684] Vgl. Kapitel 3.1.3.2.2.

[685] Vgl. Kapitel 3.1.3.2.2.

[686] Vgl. Kapitel 3.1.3.2.2.

[687] Vgl. Kapitel 3.2.2.7.2.

rungen der Sicherheit bedingen in diesem Fall eine höhere Investition in Schutzmechanismen, welche gewährleisten, dass unberechtigte Personen oder Systeme keinen Zugang erhalten. Diese erhöhten Investitionen machen sich im gesamten Entwurfsprozess, von der Anforderungserhebung über die Implementierung und Einführung bis hin zur (organisatorischen und technischen[688]) Wartung, bemerkbar.

Interoperabilität

Interoperabilität ist die Fähigkeit einer Software, mit einem oder mehreren anderen Softwaresystemen interagieren zu können.[689] Interoperabilität einer SOA wird positiv bewertet.

Zur Bewertung der Interoperabilität kann die Art der Kommunikationsmechanismen und die Art der Datenmechanismen herangezogen werden.[690] Die Art der eingesetzten *Kommunikationsmechanismen* auf der einen Seite und auf der Seite der anderen Unternehmens-IT, mit der die erste Unternehmens-IT interoperabel sein soll, hat deswegen Einfluss auf die Interoperabilität, da beide Systeme miteinander kommunizieren müssen. Der Idealfall ergibt sich hier, wenn beide Systeme, die zusammenarbeiten sollen, gleiche Kommunikationsmechanismen einsetzen und diese somit nicht angepasst werden müssen. Software auf Basis einer SOA verfügt über eine einheitliche Kommunikationsinfrastruktur.[691] Dies unterstützt die Kommunikation mit anderen Systemen. Für jeden Kommunikationsmechanismus, der von anderen IT-Systemen genutzt wird, muss eine Kommunikationsschnittstelle nur einmalig erstellt werden, weil sie der gesamten Unternehmens-IT und auch externen IT-Systemen zur Verfügung steht. Darüber hinaus ermöglicht die Verwendung einer einheitlichen Kommunikationsinfrastruktur einen zentralen Zugangspunkt für unternehmens-externe Systeme und erleichtert dadurch die Zugriffskontrolle und –authentifizierung. Eine Kommunikationsinfrastruktur unterstützt entweder die Verwendung von Standards bzw. (proprietäre) Kommunikationsmethoden eines dritten Anbieters oder eines Standardisierungsgremiums, da eine Kommunikationsinfrastruktur auf einer oder mehreren Kommunikationsmethoden aufbauen muss.

---

[688] Beispielsweise müssen Zugriffsberechtigungen auch organisatorisch gepflegt werden, wenn z. B. neue Mitarbeiter oder Geschäftspartner hinzukommen oder gehen.

[689] Vgl. Kapitel 3.1.3.2.2.

[690] Vgl. McCall /Quality factors/ S. 965.

[691] Vgl. Kapitel 2.2.4.3.

Die Verwendung von Standards erhöht die Wahrscheinlichkeit, dass andere Systeme mit der vorhandenen Kommunikationsinfrastruktur kommunizieren können. Die Verwendung einer Kommunikationsinfrastrukturlösung eines großen Drittanbieters könnte einen vergleichbaren Effekt haben. Dies muss jedoch für den konkreten Fall bewertet werden, vor allem vor dem Hintergrund, dass eine bestimmte Zielgruppe zur Kommunikation mit dem Unternehmen – z. B. die eigenen Kunden und Lieferanten – eingebunden werden soll.

Die eingesetzten *Datenmechanismen* auf Seiten der Unternehmens-IT und verbundener IT-Systeme haben einen Einfluss auf Interoperabilität. Genauso wie Kommunikationsmechanismen Einfluss auf Kommunikation und Datenaustausch haben, trifft dies auch für Datenmechanismen zu. Zur Untersuchung des Nutzenpotentials im Zusammenhang mit Interoperabilität wird in diesem Fall die Datenrepräsentation bewertet.[692] Eine SOA legt in Bezug auf Datenrepräsentationen keine Gestaltungsrichtlinie fest und beeinflusst die Wahl einer Datenrepräsentation nicht.[693] Eine direkte Auswirkung einer SOA auf Interoperabilität in Bezug auf Datenmechanismen kann deswegen nicht aufgestellt werden und die Bewertung ist neutral.

Durch die tendenziell positive Bewertung der Kommunikationsmechanismen und der neutralen Bewertung der Datenmechanismen wird die Interoperabilität insgesamt positiv bewertet.

Gehorsamkeit (Compliance)

Funktionale Gehorsamkeit bedeutet die Fähigkeit einer Software, Standards, Vereinbarungen oder Regeln (z. B. der Gesetzgebung) einhalten zu können.[694] Die funktionale Gehorsamkeit wird als neutral (mit leicht positiver Tendenz) eingestuft.

Die funktionale Gehorsamkeit einer Softwarearchitektur zu beurteilen gestaltet sich aus zwei Gründen schwierig. Zum einen kann die funktionale Gehorsamkeit als funktionale

---

[692] Vgl. McCall /Quality factors/ S. 965.

[693] Hier muss man unterscheiden von den vielfältig Aussagen, dass Standards verwendet werden sollen, und für Datenpräsentationen XML gewählt werden solle. Die Diskussion um Standards in der Softwareentwicklung ist zwar durchaus gerechtfertigt, allerdings gehört die Verwendung von bestimmten Standards wie beispielsweise XML nicht zur allgemein aufgestellten Gestaltungsrichtlinie serviceorientierter Architekturen. Wie in Kapitel 1.3.3 dargestellt können unterschiedliche technische Realisierungsmöglichkeiten verwendet werden, worunter auch Standards und proprietäre Lösungen fallen.

Anforderung ausgelegt werden, deren Erfüllung durch Softwarearchitekturen prinzipiell nicht eingeschränkt wird;[695] zum anderen unterliegen die entsprechenden Standards, Vereinbarungen und Regeln einem Wandel, weshalb im Detail nur eine Momentbewertung vorgenommen werden kann. Klammert man jedoch die Aspekte der funktionalen Gehorsamkeit aus, welche sich als funktionale Anforderung interpretieren lässt (z. B. Gesetze), kann festgestellt werden, dass sich Standards der Kommunikation, z. B. Kommunikationsprotokolle, die nach Gestaltungsprinzipien einer SOA aufgebaut sind, ändern und einbinden lassen. Das kann durch Änderungen an zentraler Stelle eines jeden Services, nämlich an der Serviceschnittstelle, und durch Änderungen an einer Stelle der Unternehmens-IT, nämlich an der Kommunikationsinfrastruktur, erfolgen. Dies ist die Begründung für die positive Bewertung der funktionalen Gehorsamkeit.

3.2.2.1.2 Nutzen der Funktionserfüllung

Bewertungen des Nutzenpotentials der Funktionserfüllung werden aus den Aspekten (1) zum Aufwand der Anforderungsanalyse, (2) zum Verständnis funktionaler Anforderungen, (3) zur Wahrung der Funktionserfüllung und (4) zur Lieferung von Echtzeitinformationen gezogen.

Aufwand der Anforderungsanalyse

Um die Dienstleistungen von Services an Geschäftsprozessen orientieren zu können,[696] müssen nicht nur die funktionalen Anforderungen der Unternehmens-IT erhoben werden, sondern darüber hinaus muss eine Analyse von Geschäftsprozessen und eine Zuordnung der funktionalen Anforderungen zu Geschäftsprozessen explizit erfolgen. Das Ergebnis dieser Analysen wird Domänenmodell genannt.[697] Der Prozess der Anforderungsanalyse mit einer anschließenden Gestaltung des Domänenmodells macht die Softwareentwicklung vor allem zu Beginn komplizierter und umfangreicher, als bei einer reinen Anforderungsanalyse mit einem darauf folgenden Grobdesign einer Software.

---

[694] Vgl. ISO /9126/ S. 8.

[695] Vgl. Kapitel 3.1.3.2.1.

[696] In dieser Arbeit wird dies als Gestaltungsziel einer SOA angesehen, vgl. Kapitel 2.2.5.2.

[697] Vgl. Kapitel 2.2.5.2.

Verständnis funktionaler Anforderungen

Dieses Vorgehen bewirkt jedoch, dass der Prozess der Anforderungserhebung intensiv durchgeführt wird und somit potentiell zu besseren Ergebnissen führen kann. Durch eine angestrebte Orientierung an Geschäftsprozessen[698] kann unterstellt werden, dass in der Phase der Anforderungserhebung ein hohes Verständnis der geforderten Funktionen erreicht wird. Diese Wirkung muss jedoch nicht zwangsläufig eintreten und stellt somit eine Tendenzaussage dar. Der umgekehrte Fall, dass eine geringere Klarheit der geforderten Anforderungen gegenüber dem Einsatz anderer Architekturkonzepte auftreten könnte, kann nicht angenommen werden. Für die Erstellung einer Grob-Analyse einer Unternehmens-IT auf Basis einer SOA muss einerseits ein Domänenmodell[699] zur Identifikation und Analyse von Geschäftsprozessen vorhanden sein und andererseits müssen zur Servicegestaltung mindestens Funktionstypen, die ein Unternehmen als Anforderung an seine Unternehmens-IT stellt, bekannt und verstanden sein. Zur Umsetzung einzelner Services ist ein umfassendes Verständnis von funktionalen Anforderungen der Geschäftsprozesse letztendlich Voraussetzung.

Wahrung der Funktionserfüllung

Ein Vorteil mit Bezug auf die Funktionserfüllung einer SOA lässt sich durch das Verfolgen des Ziels der Entkopplung im Fall der Wiederverwendung einzelner Softwarekomponenten erkennen: Durch eine hohe Entkopplung werden Dependenzen zwischen Softwarekomponenten verringert.[700] Dadurch kann Wiederverwendung erfolgen, ohne dass die Unsicherheit besteht, dass andere Softwarekomponenten in ihrer Funktionserfüllung eingeschränkt werden.

Ein weiterer Vorteil ist die Orientierung an Geschäftsprozessen. Diese Orientierung ermöglicht eine integrative Betrachtung der Unternehmens-IT und der Geschäftsprozesse eines Unternehmens.[701] Die Unternehmens-IT und die geschäftlichen Aktivitäten eines Unternehmens können somit aufeinander abgestimmt werden. Weil Geschäftsprozesse eine gemeinsame Grundlage bilden, kann eine Verringerung von Verständnisbar-

---

[698] Z. B. durch die Forderung, ein Domänen-Modell aufzustellen. Vgl. Kapitel 2.2.5.2.

[699] Vgl. Kapitel 2.2.5.2.

[700] Vgl. Sholler /Case/ S. 1; Woods /Enterprise Services Architecture/ S. 108 und META Group /Approaches/ S. 2.

rieren erreicht werden. Die IT-Abteilung kann der Fachabteilung mitteilen, was möglich ist. Die Fachabteilung kann die IT-Abteilung genauer informieren, was und in welchem Zusammenhang dies nötig ist.

Lieferung von Echtzeitinformationen

Der Genauigkeitsgrad von Informationen im Zeitverlauf ist eine nicht-funktionale Anforderung an die Funktionserfüllung. Die Ausprägung, ständig Echtzeitinformationen liefern zu können, ist die höchstmögliche Ausprägung dieses Genauigkeitsgrades. Eine Unternehmens-IT auf Basis einer SOA ermöglicht verteilte Systemzugriffe.[702] Funktionen können Daten als Echtzeitinformation abrufen, weil sie immer auf den originalen Datenbestand zugreifen.[703] Gegenüber Architekturkonzepten, bei denen kein verteilter Systemzugriff (auch über Unternehmensgrenzen hinweg) möglich ist, wird i. d. R. mit Daten gearbeitet, die nicht dem aktuellen Stand entsprechen, weil sie beispielsweise einmal pro Tag abgeglichen werden.

3.2.2.1.3 Gesamtbewertung der Funktionserfüllung

Zusammenfassend wird für das Qualitätsattribut ‚Funktionserfüllung' eine positive Bewertung festgestellt wird (Tab. 3-11).

Funktionserfüllung	***Bewertung***[704]
Hoher Aufwand der Anforderungsanalyse (Domänenmodell)	–
Hohes Verständnis der funktionalen Anforderungen	+
Wahrung der Funktionserfüllung	+
Lieferung von Echtzeitinformationen	+

[701] Vgl. zum folgenden Absatz Alshawi, Irani, Baldwin /Benchmarking/ S. 421.

[702] Vgl. Kapitel 3.2.2.7.2.

[703] Zum einen kann ein Service an dem physischen Ort ablaufen, an dem ein Datenzugriff in Echtzeit möglich ist. Der entsprechende Service wird unter Umständen über einen verteilt stattfindenden Zugriff verwendet. Zum anderen kann ein Service umgekehrt zum vorherigen Fall über einen verteilten Systemzugriff Daten abrufen und lokal mit dem Dienstleistungsnehmer in Verbindung stehen.

[704] Legende: – negativ; = neutral; + positiv; ([–,=,+]) unter bestimmten Bedingungen getroffene Argumentation.

Sicherheit	– (=)
Interoperabilität	+
Funktionale Gehorsamkeit	= (+)
Gesamtbewertung	+

Tab. 3-11: Nutzenbewertung des Qualitätsattributes Funktionserfüllung

3.2.2.2 Wandlungsfähigkeit

Wandlungsfähigkeit ist die Fähigkeit eines Softwareproduktes verändert werden zu können.[705] Änderungen können Korrekturen, Verbesserungen oder Anpassungen einer Software umfassen. Sie können durch geänderte Umweltbedingungen, Anforderungen oder funktionale Spezifikationen ausgelöst werden.

Im Folgenden werden auf Basis von Argumentationen Bewertungen der Unterqualitätsattribute der Wandlungsfähigkeit vorgenommen. Daran anschließend werden die Wirkungen einer SOA auf die Wandlungsfähigkeit einer Unternehmens-IT untersucht und es wird eine Gesamtbewertung aufgestellt.

3.2.2.2.1 Nutzenpotential der Unterqualitätsattribute der Wandlungsfähigkeit

Die Unterqualitätsattribute der Wandlungsfähigkeit sind Änderbarkeit, Erweiterbarkeit, Testbarkeit und Stabilität.[706]

Änderbarkeit

Änderbarkeit bedeutet die Fähigkeit, spezifische Änderungen an einer Software vornehmen zu können.[707] Die Änderbarkeit einer Unternehmens-IT auf Basis einer SOA wird positiv beurteilt.

In einer Unternehmens-IT, die auf Basis einer SOA beruht, sind Softwarekomponenten lose aneinander gekoppelt.[708] Dies führt zu einer Reduktion von Dependenzen zwischen

[705] Vgl. Kapitel 3.1.3.3.2.
[706] Vgl. Kapitel 3.1.3.3.2.

Servicegeber und –nehmer[709] und ermöglicht unabhängige Entwicklungszyklen.[710] Dadurch wird erreicht, dass jeder Service einen eigenen Lebenszyklus besitzt. Jeder Service kann dann unabhängig von der Anwendung, unabhängig vom Gesamtsystem und unabhängig von anderen Services gewartet und entwickelt werden.[711] Die Wandlungsfähigkeit einer Unternehmens-IT wird insofern erhöht, weil ein Potential besteht, die Softwarekomponenten einzeln verändern zu können, ohne dass Abhängigkeiten zu anderen Softwarekomponenten beachtet werden müssen.[712]

Eine SOA soll so gestaltet werden, dass Services an Geschäftsprozessen orientiert entwickelt werden. Diese Services sollen durch die Unternehmens-IT immer verwendet werden, wenn die entsprechende Dienstleistung eingesetzt werden muss. Ändert sich der entsprechende Prozess wirkt sich die entsprechende Änderung direkt auf alle Aktivitäten der Unternehmens-IT aus, die diesen Service verwenden.[713] Diese Eigenschaft unterstützt einen Software-Entwickler bei der Wartung der Unternehmens-IT, da durch die Softwarearchitektur bedingt automatisch alle Geschäftsprozesse, für die diese Änderung gültig ist, in einem Schritt umgestellt werden können. Diese Eigenschaft einer SOA kann allerdings auch zu weiteren Analyseaktivitäten führen, wenn Änderungen nur für einen Teil der Geschäftsprozesse, die eine Dienstleistung eines Services verwenden, gültig sind. In diesem Fall müssen alle entsprechenden Teile der Unternehmens-IT, die die Dienstleistung aufrufen, eine neue (versionisierte) Schnittstelle des Services aufrufen. Allerdings haben die notwendigen Aktivitäten einen geringeren Umfang als wenn alle Teile einer Unternehmens-IT einzeln angepasst werden müssen, um Änderungen durchzuführen. Durch die Abgrenzung der Services anhand von (Teilen von) Geschäftsprozessen kann die Änderbarkeit somit erhöht werden.[714]

---

[707] Vgl. Kapitel 3.1.3.3.2.

[708] Vgl. Kapitel 2.2.4.3.

[709] Vgl. Woods /Enterprise Services Architecture/ S. 108; Sholler /Case/ S. 1; META Group /Approaches/ S. 2.

[710] Vgl. Tannhäuser, Umek /Architekturmanagement/ S. 68; META Group /Approaches/ S. 2.

[711] Vgl. Sholler /Resuable Enterprise Services/ S. 1.

[712] Lederer /IT-Gesamtbankarchitektur/ S. 91.

[713] Vgl. zu den Vorteilen der Mehrfachverwendung Stahlknecht, Hasenkamp /Wirtschaftsinformatik/ S. 323.

[714] Bass, Clements und Kollegen argumentieren dies für komponentenorientierte Softwarearchitekturen. Vgl. Bass u. a. /Software architecture/ S. 55, S. 80-82, S. 105-106; Clements, Kazman, Klein /Software Architectures/ S. 118-119.

Ein Service und die dazugehörigen Serviceschnittstellen werden voneinander unabhängig implementiert werden. Servicenehmern wird diese Implementierung und somit auch die entsprechende Komplexität der Dienstleistungserstellung vorenthalten.[715] Die Wandlungsfähigkeit einer Unternehmens-IT wird dadurch unterstützt, weil Dependenzen der Implementierung nicht vorliegen.[716] Dadurch kann die Implementierung eines Services verändert werden, ohne die Interaktion mit anderen Services zu beeinflussen, so lange die Serviceschnittstelle, die die Kommunikation zum Service ermöglicht, stabil bleibt.[717] Die Stabilität von Services und Schnittstellen kann im Rahmen einer SOA dadurch erreicht werden, dass mehrere deutlich identifizierbare Entwicklungsstände von Services oder Serviceschnittstellen gleichzeitig bereitgestellt werden (sog. Versionierung).

Insgesamt ergibt sich eine positive Bewertung der Änderbarkeit.

Erweiterbarkeit

Erweiterbarkeit ist die Fähigkeit, eine Software auch nach Fertigstellung um zusätzliche Anforderungen oder um neue Datenansprüche erweitern zu können.[718] Die Erweiterbarkeit einer Unternehmens-IT auf Basis einer SOA wird positiv beurteilt.

Synonym zur Änderbarkeit kann für die Erweiterbarkeit argumentiert werden, dass lose gekoppelte Softwarekomponenten, wie sie in einer SOA vorliegen, es ermöglichen, dass jeder Service einen eigenen Lebenszyklus besitzt. Deswegen können Services unabhängig von anderen Bestandteilen einer Unternehmens-IT entwickelt werden.[719]

Die Verwendung einer einheitlichen Kommunikationsinfrastruktur, einschließlich Service-Repositorys und Service-Registrys,[720] ermöglicht eine direkte Einbindung von neuen Softwarekomponenten und von denen externer Anbieter, die die verwendete Kommunikationsinfrastruktur unterstützen. Spezielle Softwarezusätze, die einen Zugriff

---

[715] Vgl. Banke, Krafzig, Slama /Enterprise SOA/ S. 60; Natis, Schulte /Introduction/ S. 2.

[716] Vor allem in klassischen Programmen der 80er und 90er Jahre wurde beispielsweise häufig das Konzept sog. ‚globaler Variablen' eingesetzt. Hier waren Softwarekomponenten über diese global verwendbaren Variablen voneinander abhängig und solche Abhängigkeiten mussten sorgfältig beachtet und implementiert werden.

[717] Vgl. META Group /Approaches/ S. 2; Woods /Enterprise Services Architecture/ S. 26 & S. 108.

[718] Vgl. Kapitel 3.1.3.3.2.

[719] Vgl. Sholler /Resuable Enterprise Services/ S. 1.

auf die entsprechende Softwarekomponente ermöglichen, sind dafür nicht erforderlich. Neue und externe Softwarekomponenten sowie weitere Anwendungssysteme können so mit geringem Aufwand in die Unternehmens-IT integriert werden.[721]

Erweiterbarkeit kann auch anhand der Skalierbarkeit einer Unternehmens-IT untersucht werden. Skalierbarkeit beschreibt die Fähigkeit einer Unternehmens-IT, die Änderungen zu unterstützen, die durch eine starke Erhöhung der Nachfrage nach Dienstleistungen oder durch eine starke Erhöhung der Performanzansprüche hervorgerufen werden.[722] In einer SOA hat dies Einfluss (a) auf die Instanzen eines Services und der Serviceschnittstellen, (b) auf das Ausmaß oder die Form der Interaktion zwischen Services, (c) auf die Anforderungen an die Kommunikationsinfrastruktur oder (d) auf Kombinationen dieser Ausprägungen. Ein Service kann zur Erhöhung des Verarbeitungsdurchsatzes mehrmals instanziiert werden. Eine Mehrfachinstanziierung ist auch für Serviceschnittstellen vorstellbar. Im ersten Fall kann eine Lastverteilung durch die Schnittstellen geschehen, im zweiten Fall muss dies die Kommunikationsinfrastruktur unterstützen. Die Skalierung der Kommunikationsinfrastruktur, beispielsweise von Service-Repositorys, ist ebenso vorstellbar. Die praktische Umsetzungsfähigkeit einer Skalierung ist auch von der eingesetzten Technologie abhängig, da diese eine Lastverteilung und Mehrfachinstanziierung unterstützen muss. Eine solche Betrachtung wird in dieser Arbeit nicht untersucht. Als Ergebnis kann folglich festgehalten werden, dass ein Potential zur einfachen Skalierung einer SOA besteht.

Insgesamt betrachtet ergibt sich eine positive Bewertung der Erweiterbarkeit.

Testbarkeit

Testbarkeit ist die Fähigkeit, eine Software nach der Erstellung bzw. Änderung validieren zu können.[723] Es wird hierbei festgestellt, ob vorher festgelegte Kriterien bzw. Anforderungen erfüllt werden. Die Testbarkeit einer Unternehmens-IT auf Basis einer SOA wird neutral beurteilt.

---

[720] Vgl. Kapitel 2.2.4.3.

[721] Vgl. Kazman, Bass /Quality Attributes/ S. 29.

[722] Vgl. Kazman, Bass /Quality Attributes/ S. 28.

[723] Vgl. Kapitel 3.1.3.3.2.

Es gilt als belegt, dass die Kosten der Behebung von Fehlern einer Software mit der Lebensdauer der Software ansteigen.[724] Um dieses Risiko zu verringern sollte mit der Qualitätssicherung einer Software so früh wie möglich begonnen werden. Diese Vorgehensweise liegt der Annahme zugrunde, dass die entstehenden zusätzlichen Kosten der frühen Qualitätssicherung die Kosten der späteren Fehlerbehebung unterschreiten.[725] Services einer SOA stellen den Teil einer Unternehmens-IT dar, welcher einen abgegrenzten Funktionsumfang anbietet. Ein Service kann zu dem Zeitpunkt, an dem dieser fertig gestellt wurde, gegen den aufgestellten Funktionsumfang getestet werden. Weil sich der Funktionsumfang von Services an den Prozessen eines Unternehmens orientiert, werden unterschiedliche Services unterschiedliche Prozesse abdecken. Die entsprechenden Anforderungen werden hierdurch erfüllt. Eine Unternehmens-IT kann schon zu den Entwicklungszeitpunkten, zu denen ein erster Service fertig gestellt wurde, eindeutig gegen diesen Funktionsumfang getestet werden. Entsprechend der gängigen Einteilung der Testschritte in Komponenten-, Modul- und Systemtests können sowohl Komponenten- als auch Modultests frühzeitig erfolgen. Nämlich zu dem Zeitpunkt, an dem ein einzelner Service bzw. beteiligte Services fertig gestellt wurden. Ein umfassender Systemtest kann erst nach Fertigstellung aller Softwarekomponenten erfolgen, wie bei anderen Softwarearchitekturen auch.

Nachteilig für die Testbarkeit wirkt sich allerdings die lose Kopplung der Softwarekomponenten aus. Formale Definitionen der Schnittstellen sind notwendig, damit die entsprechende Dienstleistung eines Services zum einen richtige Eingabedaten erhält, zum anderen der Dienstleistungsempfänger die Rückgabedaten richtig interpretiert. Es ist erforderlich, die entsprechenden Funktionen und die aufgestellten Kommunikationsprotokolle, die für jede Schnittstelle definiert werden müssen, zu testen. Zumal Services durchaus über mehrere Schnittstellen verfügen können,[726] kann der Aufwand zum Testen der Serviceschnittstellen schnell ansteigen.

Die Testbarkeit wird insgesamt als neutral eingestuft.

---

[724] Vgl. Boehm /Software Engineering/ S. 1226-1241; Sommerville /Software Engineering/ S. 49-50.

[725] Vgl. Mellis /Projektmanagement/ S. 198-203.

[726] Vgl. Kapitel 2.2.4.2.

Stabilität

Stabilität einer Software stellt die Fähigkeit dar, unbeabsichtigte Auswirkungen durch Änderungen an der Software zu verhindern.[727] Die Stabilität einer Unternehmens-IT auf Basis einer SOA wird positiv beurteilt.

Synonym der Argumentation zur losen Kopplung bei der Änderbarkeit gilt auch für die Stabilität, dass nur einzelne Softwarekomponenten durch Änderungen betroffen sind. Infolgedessen ist die Funktionsfähigkeit von Services, die nicht auf Dienstleistungen geänderter Services zurückgreifen, keinesfalls beeinträchtigt.

Darüber hinaus bietet die Verwendung von Schnittstellen zur Kommunikation mit Services die Möglichkeit, die Dienstleistung eines geänderten Services weiterhin über die angebotene Schnittstelle zu nutzen.

Insgesamt wird die Stabilität gewährleistet und positiv bewertet.

3.2.2.2.2 Nutzenpotential der Wandlungsfähigkeit

Bewertungen des Nutzenpotentials der Wandlungsfähigkeit können auch aus Aspekten zur Schichtenbildung und zur Wandlungsdauer gezogen werden.

Schichtenbildung

‚Mehr-Schichten-Architekturen' sind Softwarearchitekturen, bei denen mehrere physische Schichten, beispielsweise Datenbankserver und Application Server, vorliegen.[728] Eine SOA ist eine ‚Mehr-Schichten-Architektur'. Es existieren mindestens die Schichten (1) Datenbanken, (2) Kommunikationsinfrastruktur, (3) Application Server, auf denen Services ablaufen, und (4) die Schicht der Präsentation (GUI). Verschiedene Autoren kommen zu dem Schluss, dass solche Architekturen eine hohe Wandlungsfähigkeit aufweisen.[729] Solche Systeme sind leicht zu rekonfigurieren, weil Softwarekomponenten sowohl als Dienstleister als auch Dienstnachfrager auftreten können und eine Verwendung schon existierender Dienstleistungen stattfinden kann. Notwendige Änderun-

---

[727] Vgl. Kapitel 3.1.3.3.2.

[728] Vgl. Zahavi /Integration/ S. 57-59.

[729] Vgl. zum folgenden Absatz Zahavie /Integration/ S. 58-59; Sommerville /Software Engineering/ S. 258-259; Gruhn, Thiel /Komponentenmodelle/ S. 5-6.

gen an einem System wirken sich nur auf die jeweiligen Schichten und auf Nutzer der Dienstleistungen dieser Schichten aus und nicht auf das gesamte System. Dies hilft, die Datenkonsistenz zu bewahren, wenn Änderungen an anderen Schichten durchgeführt werden.[730]

Dagegen ist ein Nachteil einer zu hohen Anzahl von Schichten in einer Unternehmens-IT, dass zwischen jeder Schicht eine definierte Kommunikation mit festgelegten Datentransferobjekten und definierten Schnittstellen existieren muss. Wird die Anzahl der Schichten zu hoch, resultiert daraus eine hohe Anzahl zu pflegender Schnittstellen und der entsprechenden Datenkontainer bzw. Datendefinitionen dieser Schnittstellen. Eine SOA legt die Anzahl der zu bildenden Softwareschichten auf niedrigem Niveau fest, weil mit der Ebene der Kommunikationsinfrastruktur und der Ebene der Services zwei Schichten eingefügt werden. Ein solches Vorgehen ist allerdings nicht fest vorgeschrieben und kann in der Praxis zugunsten einer klaren Strukturierung von Services umgangen werden.[731]

Da eine Schichtenarchitektur positiv eingestuft wird und eine zu hohe Schichtenbildung bei dem Einsatz eines SOA vermieden werden kann, wird dieser Teilaspekt positiv bewertet.

Wandlungsdauer

Die Wandlungsfähigkeit einer Unternehmens-IT kann nach der Wandlungsdauer beurteilt werden, die im produktiven Einsatz auftritt, um einen Fehler zu lokalisieren und zu beheben.[732] Der Einsatz einer SOA erhöht die Komplexität der Gesamtstruktur einer Unternehmens-IT. Zum einen liegt eine Mehr-Schichten-Architektur vor, in der Fehler in jeder Schicht auftreten können; zum anderen müssen unterschiedliche Softwarekomponenten bei der Bearbeitung einzelner Geschäftsprozesse zusammenarbeiten. Die Erhöhung der Komplexität zur Aufgabenbearbeitung wirkt sich nachteilig auf die Zeitdauer der Wandlung aus, da komplexe Beziehungen und Abhängigkeiten beachtet werden müssen. Ein Service kann z. B. nicht ohne weiteres entfernt werden, weil die Gefahr

---

[730] Vgl. Raasch /Systementwicklung/ S. 32.

[731] Wie in Kapitel 2.2.4.1 dargestellt, kann beispielsweise zwischen Infrastrukturservices und Services unterschieden werden.

[732] Raasch /Systementwicklung/ S. 31.

besteht, dass dienstleistungsnehmende Services ihre entsprechenden Aufgaben nicht mehr in vollem Umfang bzw. entsprechender Qualität erfüllen können.

Andererseits kann auch eine positive Wirkung auf die Wandlungsdauer identifiziert werden: Durch eine Unterteilung einer Unternehmens-IT in Module wird die Anzahl der der durch Fehler und entsprechend notwendige Änderungen betroffenen Funktionen reduziert.[733] Für eine SOA bedeutet dies, dass durch die eindeutige Zuordnung von Funktionen auf Services der Service schnell identifiziert werden kann, dessen Dienstleistung fehlerhaft ist. Außerdem wird die Behebung eines Fehlers unterstützt, weil Services lose gekoppelt sind.[734] Abhängigkeiten der Implementation sind nicht bzw. nur in sehr geringem Ausmaß vorhanden. Beide Zeitfaktoren zur Fehlerbeseitigung, nämlich die Fehlerlokalisation und die Fehlerbehebung, können somit potentiell verringert werden.

Ein weiterer Aspekt, der die Wandlungsdauer positiv beeinflusst, ist das Potential, Änderungen an Funktionen auch dann schnell umsetzen zu können, wenn bei Änderung der Implementation auch eine Erweiterung/Änderung bestehender Schnittstellen oder die Neuimplementation weiterer Schnittstellen notwendig werden. Der Aufwand für die notwendige Implementation von spezialisierten Schnittstellen für jeden einzelnen Dienstleistungsnehmer entfällt, weil die Dienstleistung in ein festgelegtes Format über einen Kanal und eine definierte Schnittstelle allen Dienstleistungsnachfragern angeboten wird. Es besteht allerdings die Unsicherheit, dass das Design der Schnittstellen so viele Ressourcen zur Erfüllung der Anforderungen aller nachfragenden Dienstleistungsnehmer beansprucht, dass sich dieser potentielle Vorteil auch negativ auswirken kann.[735]

Ein umgekehrtes Bild ergibt sich, wenn der Funktionsumfang einzelner Services so umfassend gewählt wird, dass komplexe Geschäftsprozesse durch einen Service abgedeckt werden. Die zu beachtende Komplexität verringert sich. Erweiterungen können dann unter Beachtung einer geringeren Anzahl von Beziehungen und Abhängigkeiten umge-

---

[733] Vgl. Bass u. a. /Software architecture/ S. 55, S. 80-82, S. 105-106; Clements, Kazman, Klein /Software Architectures/ S. 118-119.

[734] Abhängigkeiten der Implementation sind dann gegeben, wenn Services eine geringe lose Kopplung aufweisen. Vgl. Kapitel 2.2.4.3.

[735] An dieser Stelle sei auf die Diskussion dieser Problematik in Kapitel 4.3.2 hingewiesen.

setzt werden. Dagegen erhöht sich potentiell der Zeitbedarf der Fehlerlokalisierung und Fehlerbehebung.

Insgesamt werden die Auswirkungen einer SOA auf die Wandlungsdauer neutral bewertet.

3.2.2.2.3 Gesamtbewertung der Wandlungsfähigkeit

Zusammenfassend wird für das Qualitätsattribut ‚Wandlungsfähigkeit' eine positive Bewertung festgestellt (Tab. 3-12).

(Unter-)Qualitätsattribute	*Wirkung*
Schichtenanzahl	+
Wandlungsdauer	=
Änderbarkeit	+
Erweiterbarkeit	+
Testbarkeit	=
Stabilität	+
Gesamtbewertung	+

Tab. 3-12: Nutzenbewertung des Qualitätsattributes Wandlungsfähigkeit

3.2.2.3 Benutzbarkeit

Benutzbarkeit ist die Fähigkeit einer Software, den Software-Entwickler darin zu unterstützen, eine gesamte Software oder nur Teile davon zur Entwicklung verwenden zu können.[736]

Im Folgenden werden Bewertungen der Unterqualitätsattribute der Benutzbarkeit vorgenommen. Daran anschließend werden die Wirkungen einer SOA auf die Benutzbarkeit einer Unternehmens-IT untersucht und eine Gesamtbewertung wird aufgestellt.

[736] Vgl. Kapitel 3.1.3.4.2

3.2.2.3.1 Nutzenpotential der Unterqualitätsattribute der Benutzbarkeit

Die Unterqualitätsattribute der Benutzbarkeit sind Verständlichkeit, Betriebsfähigkeit, Kommunikationsfähigkeit und Gehorsamkeit.[737]

Verständlichkeit

Verständlichkeit stellt die Fähigkeit einer Software dar, dem Entwickler eindeutig verständlich zu machen, was die Software leisten und wie diese für eine zu erfüllende Aufgabe eingesetzt werden kann.[738] Je kürzer die Zeit, die benötigt wird, um eine Software zu verstehen, desto verständlicher ist diese. Die Verständlichkeit einer Unternehmens-IT auf Basis einer SOA wird positiv beurteilt.

Eine SOA setzt eine Kommunikationsinfrastruktur zur Unterstützung der Kommunikation der Services untereinander ein.[739] Die Kommunikationsinfrastruktur beinhaltet Mechanismen wie ein Service-Repository, ein Service-Registry und einen Service-Bus. In dem Service-Repository werden die Informationen jedes Services gespeichert, die anderen Services Auskunft über die Leistungen des Services geben. Ein Service-Registry unterstützt den Prozess der Suche nach angebotenen Dienstleistungen in einem Service-Repository. Diese Funktion einer Kommunikationsinfrastruktur einer SOA kann durch Software-Entwickler ebenso zur Informationsbeschaffung genutzt werden. Weil diese Informationen an zentraler Stelle für die gesamte Unternehmens-IT abgerufen werden kann, verringert sich der Zeitaufwand, um an die entsprechenden Informationen zu gelangen. Das Vorhandensein und die Richtigkeit der Informationen ist die Voraussetzung zur Funktionsfähigkeit einer Unternehmens-IT auf Basis einer SOA. Demzufolge kann sich ein Entwickler auf das Vorliegen und auf die Richtigkeit der Informationen verlassen. Die Kommunikationsinfrastruktur ist somit nicht nur für das Funktionieren einer Unternehmens-IT auf Basis einer SOA notwendig, sondern unterstützt den Software-Entwickler auch bei seiner Arbeit. Sie erläutert die Funktion von Services und unterstützt die Suche nach spezifischen Funktionen bzw. Services. Die Verständlichkeit der Services kann weiter erhöht werden, wenn ein Service-Repository um zum Betrieb nicht zwingend notwendige Informationen ergänzt wird. Beispielsweise können dies Informa-

---

[737] Vgl. Kapitel 3.1.3.4.2.

[738] Vgl. Kapitel 3.1.3.4.2.

[739] Vgl. Kapitel 2.2.4.3.

tionen zum Entwicklungsstand, zur Versionierung, zur organisatorischen Verantwortlichkeit und Ähnliches sein.

Durch die Verwendung einer Kommunikationsinfrastruktur für die gesamte Unternehmens-IT wird ein unternehmensweit einheitliches Kommunikationsprinzip festgelegt. Dieses kann zwar in den jeweiligen Ausprägungen von einer zur anderen Unternehmens-IT unterschiedlich sein.[740] Jedenfalls wird dadurch erreicht, dass interne Kommunikationsabläufe schneller verstanden werden können als wenn die Gestaltung für die Kommunikation einer spezifischen Anwendung zunächst festgestellt werden muss. Vermieden wird dadurch auch die Anforderung an Software-Entwickler, mehrere unterschiedliche Kommunikationsarten innerhalb eines Unternehmens verstehen, anwenden und pflegen zu müssen. Dies erhöht abermals das Verständnis der Unternehmens-IT.

Wird die Orientierung an Geschäftsprozessen konsequent verfolgt und werden Choreographien[741] von Services anhand der Geschäftsprozessanalyse zur Anforderungserhebung durchgeführt, findet kein Wechsel der Präsentationen zwischen Analyse und Orchestrierung von Services statt. Dies führt zu einer höheren Verständlichkeit der Software und unterstützt den Entwicklungsprozesses.[742]

Weil in einer Unternehmens-IT auf Basis einer SOA mehrere abgrenzbare Entwicklungsstände von Services oder Serviceschnittstellen gleichzeitig bereitgestellt werden können (sog. Versionierung), besteht das Risiko, dass die Verständlichkeit, wie und durch wen eine Dienstleistung erbracht wird, verringert wird. Durch die Verwendung einer Kommunikationsinfrastruktur wird dieses Risiko jedoch minimiert, da sie die Verständlichkeit bezüglich verschiedener Versionen erhöht. Eine Kommunikationsinfrastruktur bildet eine Eindeutigkeit über Unterschiede zwischen Versionen ab und darüber, welche Version wo verwendet wird bzw. verwendet werden soll; sie kann sogar um Informationen ergänzt werden, welche Version von welchen dienstleistungsnehmenden Services verwendet wird.

Durch das Verbergen von Komplexität einzelner Services wird die Komplexität von einem Service, der andere Services als Dienstnehmer in Anspruch nimmt, verringert. Die entsprechende Funktion eines Services steht dem dienstleistungsnehmenden Service

---

[740] Beispielsweise durch den Einsatz unterschiedlicher Technologien.

[741] Vgl. Kapitel 2.2.5.2.

als in sich geschlossene Funktion zur Verfügung und muss nicht durch den Software-Entwickler identifiziert werden.

Diese Möglichkeit der Orchestrierung von Services steigert (erwartungsgemäß) die strukturelle Komplexität einer Unternehmens-IT, weil eine Verschachtelung von Services zur Dienstleistungserbringung notwendig wird. Dies reduziert tendenziell die Verständlichkeit einer Unternehmens-IT.

Die Verständlichkeit des Quellcodes wird nicht durch eine SOA beeinflusst, da diese keine Gestaltungsrichtlinien für die Ebene der Programmiersprachen vorgibt. Auswirkungen auf die Verständlichkeit des Quellcodes haben vielmehr Vorgaben determinierender Dokumentation.[743] Betrachtet man die Gestaltungsrichtlinien, die Einfluss auf Quellcodes haben könnten, kann dennoch keine eindeutige Argumentation zugunsten einer positiven oder negativen Bewertung gefunden werden. Es lassen sich sowohl Verbesserungen (durch Servicebeschreibung, definierte Kommunikationsmethoden, Beibehaltung von Präsentationen, Unterstützung der Phase der Orchestrierung und die Kapselung von Funktionen) als auch Verschlechterungen der Verständlichkeit (durch Versionierung von Services, von Schnittstellen und eine Erhöhung struktureller Komplexität) finden.

Weil sich lediglich ein marginales Übergewicht einer positiven Bewertungsrichtung ergibt, wird insgesamt eine neutrale Bewertung gezogen.

Betriebsfähigkeit

Betriebsfähigkeit ist die Fähigkeit einer Software, den Software-Entwickler in der Anwendung und Kontrolle der Software zu unterstützen.[744] Die Betriebsfähigkeit wird neutral eingestuft.

---

[742] Vgl. Raasch /Systementwicklung/ S. 430-432.

[743] Determinierende Dokumentation stellt Vorgaben für Programmierer auf, wie Software entwickelt werden soll. Vgl. Ambler /Modeling/ S. 9; Boehm nennt den determinierenden Einsatz von Dokumentation „Vorabdokumentation“ und erstellt diese, „um [...] Ziele und Pläne für künftige Tätigkeiten der Softwareentwicklung zu definieren [...]“, Boehm /Software-Produktion/ S. 37-38; Determinierende Dokumentation kann auch „Projektvorgaben“ darstellen, vgl. Müller /Software-Engineering/ S. 49. Beispiele determinierender Dokumentation sind Namenskonventionen, Formatierungs- und Dokumentationsrichtlinien.

[744] Vgl. ISO /9126/ S. 9; McCall /Quality factors/ S. 965.

In einer Unternehmens-IT auf Basis einer SOA sind Services i. d. R. voneinander und von anderen Systemen und Anwendungen unabhängig.[745] Wenn Services so orchestriert werden, dass diese aufeinander aufbauen, um eine Dienstleistung zu erbringen, ist die Erbringung der Dienstleistung von den Services abhängig. Ist diese Dienstleistung in einem Service implementiert, ist diese folglich von den dienstleistungserbringenden Services abhängig. Weil ein einheitlicher Funktionsumfang in dieser Arbeit nicht für alle Services einer Unternehmens-IT und folglich auch nicht für Services aller SOA festgelegt werden kann,[746] ist eine eindeutige Aussage der Abhängigkeit von Services nicht möglich. Tendenziell sind Services von einem großen Anteil aller anderen Services einer Unternehmens-IT unabhängig, da ein Service in einer Orchestrierung nicht auf alle dienstleistungsanbietenden Services zurückgreifen wird. Die Unabhängigkeit eines großen Teils der Services einer Unternehmens-IT auf Basis einer SOA führt dazu, dass diese Services auch dann funktionsfähig sind, wenn ein anderer Service ausfällt. Das Ausmaß eines Ausfalls eines Services ist demzufolge begrenzt.

Die Betriebsfähigkeit einer Unternehmens-IT ist stark von der Betriebsfähigkeit der verwendeten Infrastruktur abhängig. Bei einer SOA lässt sich feststellen, dass wichtige zentrale Stellen bestehen, deren Ausfall die Betriebsfähigkeit der gesamten Unternehmens-IT stark beeinflussen würde. Darunter fallen z. B. alle Bestandteile der Kommunikationsinfrastruktur. Für den konkreten Fall muss die eingesetzte technische Lösung der Kommunikationsinfrastruktur bewertet werden.[747]

Die Betriebsfähigkeit wird insgesamt neutral eingestuft.

Kommunikationsfähigkeit

Kommunikationsfähigkeit bedeutet, dass die Software nützliche Eingabe- und Ausgabeformate zur Kommunikation verwendet.[748] Hierunter fallen auch die Aspekte des Ein-/Ausgabevolumens und der Ein-/Ausgaberate,[749] welche eine Anwendung der Software im geplanten Umfang erlauben sollten.

---

[745] Vgl. Natis /SOA/ S. 23.

[746] Vgl. Kapitel 2.2.4.1.

[747] Eine solche Bewertung wird jedoch in dieser Arbeit nicht durchgeführt. Vgl. Kapitel 1.3.3.

[748] Vgl. McCall /Quality factors/ S. 965.

[749] Vgl. McCall /Quality factors/ S. 965.

Services kommunizieren über eine Kommunikationsinfrastruktur miteinander.[750] Die Verwendung einer konkreten Kommunikationsinfrastruktur beeinflusst die Wahl der Kommunikationsformate zur Datenübermittlung. Die Wahl der Kommunikationsformate zur Ein- und Ausgabe ist jedoch unabhängig davon.[751] Die Kommunikationsformate können dementsprechend definiert werden, sodass eine bestmögliche Unterstützung der Unternehmens-IT stattfindet.

Die Kommunikationsinfrastruktur kann zur Informationsbeschaffung auch durch den Software-Entwickler genutzt werden. Diese hilft ihm, die gesuchten Services anhand ihres im Service-Repository hinterlegten Funktionsumfangs zu finden. Eine Analyse der Kommunikation über die Kommunikationsinfrastruktur liefert darüber hinaus auch Informationen zu Beziehungen zwischen Services auf unternehmensweiter Ebene und zu Verwendungshäufigkeiten einzelner Services. Die gesamte Kommunikation zwischen Services fließt über die Kommunikationsinfrastruktur, wodurch Auswertungen über das Laufzeitverhalten von Services und Ähnlichem möglich ist. Die Kommunikationsfähigkeit einer SOA, Informationen zur Unternehmens-IT und einzelnen Services zu liefern, wird deswegen als positiv eingestuft.

Services bieten deutlich abgegrenzte Funktionen als Dienstleistung an, die durch andere Services aufgerufen werden können. Mit jeder Nutzung eines Services ist auch eine Kommunikation über die Kommunikationsinfrastruktur notwendig und ggf. sogar jedes Mal die Nutzung aller Bestandteile der Kommunikationsinfrastruktur, wie z. B. die Befragung eines Service-Repositorys. Diese intensive Nutzung der Kommunikationsinfrastruktur erhöht das Kommunikationsaufkommen und verringert tendenziell die Ein- und Ausgaberate der Unternehmens-IT. Die Kommunikationsinfrastruktur, insbesondere das Service-Repository und das Service-Registry, ist dadurch sehr stark beansprucht und kann tendenziell einen Engpass zur Kommunikation zwischen Services darstellen.

Weil die gesamte Kommunikation einer Unternehmens-IT über die Kommunikationsinfrastruktur durchgeführt wird, ist auch das Kommunikationsvolumen der Unternehmens-IT von der Kommunikationsinfrastruktur abhängig. Hier kann zwischen Brutto- und Nettovolumen unterschieden werden. Das Nettovolumen stellt das Datenvolumen

---

[750] Vgl. Kapitel 3.1.3.4.2

dar, welches zur Durchführung einer Dienstleistung notwendig ist und als Ergebnis einer Dienstleistung ausgegeben wird; zum Bruttovolumen zählt auch das Datenvolumen, welches zur Steuerung der Kommunikation, zur Auffindung eines Dienstleistungsanbieters und zur Kommunikation mit der Kommunikationsinfrastruktur (z. B. zur Übermittlung einer aktuellen Adresse, unter der ein Service gefunden werden kann) aufgewendet werden muss. In einer SOA übersteigt das Bruttovolumen das Nettovolumen unter Umständen stark, weil Services intensiv mit der Kommunikationsinfrastruktur kommunizieren müssen, um Dienstleistungsanbieter zu finden und um mit diesen in Verbindung treten zu können.

Zusammenfassend wird die Bewertung der Kommunikationsfähigkeit neutral beurteilt.

3.2.2.3.2 Nutzenpotential der Benutzbarkeit

Durch die Nutzung von Schnittstellen, die eindeutig einem Service zugeordnet sind, und durch die Möglichkeit der Versionierung von Schnittstellen kann eine Schnittstelle sehr lange betrieben werden. Selbst eine Erweiterung oder eine Änderung an der Funktion eines Services machen es nicht zwingend notwendig, eine Schnittstelle zu modifizieren. Diese Stabilität der Schnittstellen ermöglicht, dass Services auf bekannte Art und Weise von Software-Entwicklern bei der Entwicklung von Services und bei der Orchestrierung zur Abbildung von Prozessen der Unternehmens-IT angesprochen werden. Dies wiederum erhöht die Benutzbarkeit, weil die Verwendung einer Dienstleistung über bestehende Schnittstellen erfolgen kann.

Durch die Möglichkeit, unterschiedliche Schnittstellen anbieten zu können, können Dienstleistungen eines Services über diejenigen Formate und Eingabeparameter abgerufen werden, die für die Nutzung der Dienstleistung im konkreten Fall adäquat sind. Die Benutzbarkeit für Software-Entwickler vergrößert sich dadurch, weil spezielle Formatkonvertierungen, die von einem dienstleistungsgebenden Service vorausgesetzt werden, nicht erforderlich sind. Auch entfällt die Notwendigkeit, Eingabeparameter angeben zu müssen, die für den konkreten Fall nicht relevant sind, aber von dem dienstleistungsan-

---

[751] Ggf. wird die Wahl eines Kommunikationsformats zur Ein- und Ausgabe dadurch beeinflusst, dass zusätzliche Dienstleistungen wie beispielsweise Konvertierungen zwischen unterschiedlichen Formaten durch die Kommunikationsinfrastruktur angeboten werden.

bietenden Service in bestimmten Fällen zur korrekten Funktionserfüllung vorausgesetzt werden.

3.2.2.3.3 Gesamtbewertung der Benutzbarkeit

Zusammenfassend wird das Qualitätsattribut ‚Benutzbarkeit' positiv bewertet (Tab. 3-13).

(Unter-)Qualitätsattribute	*Wirkung*
Stabilität von Schnittstellen	+
Gebrauchsgerechte Schnittstellengestaltung	+
Verständlichkeit	=
Betriebsfähigkeit	=
Kommunikationsfähigkeit	=
Gesamtbewertung	+

Tab. 3-13: Nutzenbewertung des Qualitätsattributes Benutzbarkeit

3.2.2.4 Verlässlichkeit

Verlässlichkeit ist die Fähigkeit einer Software, die es erlaubt, volles Vertrauen in ihre Dienstleistungen zu haben, weil spezifizierte Ausprägungen der zu erbringenden Dienstleistung unter spezifizierten Bedingungen eingehalten werden.[752]

Im Folgenden werden auf Basis von Argumentationen Bewertungen der Unterqualitätsattribute der Verlässlichkeit vorgenommen. Anschließend wird die Wirkung einer SOA auf die Verlässlichkeit einer Unternehmens-IT untersucht und eine Gesamtbewertung aufgestellt.

[752] Vgl. Kapitel 3.1.3.5.2.

3.2.2.4.1 Nutzenpotential der Unterqualitätsattribute der Verlässlichkeit

Die Unterqualitätsattribute der Verlässlichkeit sind Robustheit, Wiederherstellbarkeit, Gehorsamkeit und Verfügbarkeit.[753]

Robustheit

Robustheit stellt die Fähigkeit einer Software dar, auch dann noch korrekt[754] zu funktionieren, wenn (1) Eingabeparameter nicht den durchgeführten Testeingaben entsprechen, wenn (2) Fehlbedienungen einer Software erkannt und z. B. durch entsprechende Meldungen an den Nutzer bearbeitet werden und (3) die Software nach dem Auftreten dieser Fälle weiterhin genutzt werden kann.[755]

Generell bietet eine SOA die Möglichkeit, eine robuste Unternehmens-IT zu implementieren. Einschränkungen, warum eine Software mit nicht getesteten Eingabeparametern aufgrund der Nutzung einer SOA nicht funktionieren sollte, können nicht identifiziert werden.

Auch die Erkennung einer Fehlbedienung wird durch eine SOA nicht signifikant beeinflusst. Hier ergibt sich u. U. nur die Notwendigkeit, dass eine Fehlbedienung erst auf einer tiefen Ebene festgestellt wird und über alle beteiligten Services wieder zurückgegeben werden muss. Außerdem muss die Instanz einer Choreographie von Services in der Lage sein, ihren Ablauf zu unterbrechen, um den Nutzer auf eine Fehleingabe oder Fehlfunktion aufmerksam machen zu können. Die weitere Funktionsfähigkeit von den sowohl an einer unterbrochenen Choreographie beteiligten und als auch von den nicht daran beteiligten Services bleibt dabei unbeeinflusst. Die Unternehmens-IT kann deswegen nach Auftreten von Fehlern weiter genutzt werden.

Eine SOA besitzt aber im Hinblick auf die Robustheit ein kritisches Gestaltungselement: die Kommunikationsinfrastruktur. Diese kann aus vielfältigen technischen Lösungsalternativen aufgebaut werden, weshalb nicht auf die einzelnen möglichen Lösungen eingegangen wird. Erwähnenswert jedoch ist, dass ein Ausfall einer Komponente der Kommunikationsinfrastruktur, z. B. des Service-Busses oder des Service-

---

[753] Vgl. Kapitel 3.1.3.5.2.

[754] Korrektes Funktionieren beschreibt einen Ablauf gemäß spezifizierten Ausprägungen unter spezifizierten Bedingungen.

Repositorys, zu einem Zustand der Unternehmens-IT führen kann, in dem diese insgesamt nicht mehr funktionsfähig ist.

In einer Unternehmens-IT auf Basis einer SOA können Services auch über Unternehmensgrenzen hinweg angeboten und genutzt werden. Wird ein Service einer Unternehmens-IT durch einen Dritten genutzt, so besteht die Gefahr, dass Fehleingaben durch das fremde System auftreten. Durch das Zulassen externer Nutzer eines Services müssen somit erhöhte Maßnahmen der Sicherheitsprüfungen und der Eingabedatenvalidität ergriffen werden. Wenn externe Services genutzt werden, ist es notwendig, die Rückgaben des externen Services zu überprüfen. Bei Verdacht auf unrichtige oder manipulierte Rückgaben wird eine Weiterverarbeitung gestoppt. Das Erlauben von Zugriffen von und auf externe Systeme macht es nötig, Authentifizierungsmechanismen umzusetzen und zu pflegen. Dadurch wird verhindert, dass von außerhalb gezielt Falschinformationen eingegeben werden oder Manipulationen durchgeführt werden können, die den Zusammenbruch der Unternehmens-IT zum Ziel haben.

Insgesamt wird die Robustheit einer SOA neutral beurteilt.

Wiederherstellbarkeit

Wiederherstellbarkeit ist die Fähigkeit einer Software, einen spezifizierten Dienstleistungszustand (z. B. ein Performanzlevel oder Datenintegrität) zu erreichen, nachdem ein fehlerhafter Zustand eingetreten ist.[756]

Durch die lose Kopplung der Services[757] ist es möglich, ausgefallene oder nicht mehr erreichbare Services auf Ausweichcomputern ablaufen zu lassen, z. B. weil ein Computer oder Netzwerk durch einen Hardwarefehler ausgefallen ist. Die neue Adresse, unter der ein verschobener Service erreichbar ist, ist der gesamten Unternehmens-IT durch die Kommunikationsinfrastruktur bekannt, und der entsprechende Service kann weiter genutzt werden. Durch Überprüfungsmechanismen, die sich in eine Kommunikationsinfrastruktur integrieren lassen, ist eine solche Instanziierung eines Services an neuer Stelle sogar automatisch erfüllbar.

---

[755] Vgl. Kapitel 3.1.3.5.2.

[756] Vgl. Kapitel 3.1.3.5.2.

[757] Vgl. Kapitel 2.2.4.3.

Gehorsamkeit

Gehorsamkeit in Bezug auf die Verlässlichkeit bedeutet die Fähigkeit einer Software, entsprechende Standards, Vereinbarungen oder Regelungen (z. B. Gesetze) mit Bezug zur Verlässlichkeit einhalten zu können.[758]

Eine SOA bestimmt, wie Kommunikation innerhalb einer Unternehmens-IT durchzuführen ist und wie Services miteinander in Beziehung stehen. Die Einhaltung von Standards, Vereinbarungen oder sonstiger Regeln kann dadurch beschränkt sein. Beispielsweise kann bei einer asynchronen Servicebeziehung ein Standard, der eine zeitnahe und synchrone Bearbeitung einer Aufgabe fordert, nicht zwingend eingehalten werden. Für einen solchen Fall müssten in einer SOA weitere Maßnahmen ergriffen werden, damit entsprechende Standards, Vereinbarungen oder sonstige Regeln entsprechend ihrer Vorgaben eingehalten werden können. Die Einführung einer Maßnahme zur Einhaltung von Standards usw. ist mit zusätzlichem Aufwand zur Implementierung verbunden und muss darüber hinaus durch die Kommunikationsinfrastruktur unterstützt werden, weil diese ebenso entsprechenden Anforderungen unterworfen ist.

Die Gehorsamkeit wird negativ bewertet.

Verfügbarkeit

Verfügbarkeit ist die Fähigkeit einer Software, Dienste dann an den Nutzer zu liefern, wenn diese nachgefragt werden.[759]

Um eine Dienstleistung in einer Unternehmens-IT, die auf Basis einer SOA aufbaut, anbieten und nutzen zu können, müssen Nachfrager mehrere Schritte vollziehen. Zuerst muss eine Anfrage an die Kommunikationsinfrastruktur (im Speziellen an das Service-Registry oder Service-Repository) gestellt werden, ob die gewünschte Dienstleistung abrufbar ist. Darauf folgend müssen dann Informationen von der Kommunikationsinfrastruktur geliefert werden, wie und unter welchen Bedingungen die Dienstleistung abgerufen werden kann. Anschließend muss der Dienstnachfrager die Dienstleistung über einen Service abrufen und nutzt dazu wieder die Kommunikationsinfrastruktur. Die Verfügbarkeit der gesamten Unternehmens-IT ist somit sehr stark von der Verfügbarkeit

---

[758] Vgl. Kapitel 3.1.3.5.2.

[759] Vgl. Kapitel 3.1.3.5.2.

der Kommunikationsinfrastruktur abhängig. Diese wird extrem belastet, weil jede Dienstleistungsnachfrage – wie beschrieben – eine intensive Nutzung der Kommunikationsinfrastruktur darstellt.

Die Verfügbarkeit der Unternehmens-IT ist aber auch abhängig von der Verfügbarkeit der einzelnen Funktionen, die in einer SOA in Services gekapselt sind. Folglich hängt die Verfügbarkeit der Unternehmens-IT von der Verfügbarkeit der einzelnen Services ab. Mit der Möglichkeit, mehrere Instanzen eines Services auch verteilt in einer Unternehmens-IT ablaufen zu lassen und bei Ausfall eines Services ggf. automatisch einen Ersatz-Service instanziieren zu können,[760] kann hier von einer hohen Verfügbarkeit der einzelnen Services ausgegangen werden.

Insgesamt wird die Verfügbarkeit neutral bewertet.

3.2.2.4.2 Nutzenpotential der Verlässlichkeit

Eine SOA unterstützt das Konzept verteilter Systeme.[761] Es besteht die Möglichkeit, eine Unternehmens-IT aufzubauen, in der die Erfüllung einer Dienstleistungsnachfrage unabhängig von der Erbringung der Dienstleistung stattfinden kann. Durch eine solche Strukturierung einer Unternehmens-IT werden Probleme vermieden, die bei einer gleichartigen Anwendungswartung und redundanten Datenhaltung auftreten können.[762] Gleichartige Anwendungswartung ist erforderlich, wenn eine Unternehmens-IT an verschiedenen Standorten gleiche Prozesse in unterschiedlichen Softwarekomponenten implementiert hat und durch notwendig gewordene Änderungen alle Softwarekomponenten umgewandelt werden müssen. Redundante Datenhaltung kommt vor, wenn Daten an unterschiedlichen Standorten nachgefragt werden, aber keine direkte Möglichkeit zum Datenaustausch zwischen Unternehmen besteht bzw. nicht in Echtzeit erfolgen kann. Die damit verbundenen Probleme sind z. B. (a) unterschiedliche Umsetzungen von Änderungen, (b) eine inkonsistente Prozessdurchführung mit eventuell unterschiedlichen Arbeitsergebnissen, oder (c) eine gleichzeitig stattfindende, aber widersprüchli-

---

[760] In Kapitel 3.2.2.2.1 wurde gezeigt, dass eine Unternehmens-IT auf Basis einer SOA gut skalierbar ist.

[761] Vgl. Kapitel 3.2.3.3.1.

[762] Vgl. Kaib /EAI/ S. 24-25; Raasch /Systementwicklung/ S. 133-134.

che Bearbeitung von Daten.[763] Weil diese Probleme vermieden werden können, wird die Anforderung an eine Unternehmens-IT durch eine SOA unterstützt, Vertrauen in konsistente Daten, Prozesse und Arbeitsergebnisse schaffen zu können.

Die Verlässlichkeit einer Unternehmens-IT wird auch dadurch erhöht, wenn Fehler in der Software so schnell wie möglich behoben werden, damit das Vertrauen in die nachgefragte und spezifizierte Dienstleistung wiederhergestellt wird. Die Eigenschaft einer SOA, Services lose zu koppeln, ermöglicht es den Software-Entwicklern, einen eigenen Lebenszyklus für jeden Service umzusetzen. Dies bedeutet, dass die Fehlerbehebung in einem Service unabhängig von anderen Services geschehen und ein korrigierter Service sofort verwendet werden kann. Diese Eigenschaft einer SOA ist auch bei geplanten Updates einer Unternehmens-IT von Vorteil, da erstens Verbesserungen einer Funktion zeitnah in der Produktion eingesetzt werden können und zweitens Auswirkungen auf andere Services durch die lose Kopplung minimal sind. Durch Service- und Schnittstellenversionisierung ist es sogar möglich, ein Serviceupdate ohne Auswirkung auf andere Services durchzuführen, weil bestehende Dienstleistungsnehmer ggf. explizit auf die bisherige Serviceversion zurückgreifen können.

3.2.2.4.3 Gesamtbewertung der Verlässlichkeit

Zusammenfassend wird das Qualitätsattribut ‚Verlässlichkeit' positiv bewertet (Tab. 3-14)

(Unter-)Qualitätsattribute	*Wirkung*
Sicherstellung konsistenter und spezifizierter Abläufe	+
Sicherstellung konsistenter Daten	+
Wiederherstellung der Verlässlichkeit	+
Robustheit	(+)
Wiederherstellbarkeit	+
Gehorsamkeit	–

[763] Vgl. Raasch /Systementwicklung/ S. 133-134.

Verfügbarkeit	=
Gesamtbewertung	+

Tab. 3-14: Nutzenbewertung des Qualitätsattributes Verlässlichkeit

3.2.2.5 Effizienz

Effizienz ist die Fähigkeit einer Software, ihre festgelegte Funktionalität mit einem Minimum an einzusetzenden Ressourcen unter festgelegten Bedingungen bereitzustellen.[764]

Im Folgenden werden auf Basis von Argumentationen Bewertungen der Unterqualitätsattribute der Effizienz vorgenommen. Daran anschließend werden die Wirkungen einer SOA auf die Effizienz einer Unternehmens-IT untersucht und es wird eine Gesamtbewertung aufgestellt.

3.2.2.5.1 Nutzenpotential der Unterqualitätsattribute der Effizienz

Die Unterqualitätsattribute der Effizienz sind Antwortzeitverhalten, Grad der Ressourcenausnutzung und Gehorsamkeit.[765]

Antwortzeitverhalten

Das Antwortzeitverhalten gibt die Fähigkeit der Software an, möglichst geringe Antwort- und Verarbeitungszeiten sowie gleichzeitig hohe Durchsatzraten zu erzielen, wenn Funktionen unter festgelegten Bedingungen ausgeführt werden.[766]

Aufgrund der losen Kopplung der Services untereinander und der daraus resultierenden Notwendigkeit der Kommunikation mit einem Service, sobald die Funktionen des Services verwendet werden sollen, müssen Informationen von dem Service (bzw. von seinen Serviceschnittstellen) angenommen, verarbeitet und versendet werden können. Eine Annahme zur Verarbeitung einer Information beinhaltet, dass der Inhalt der Information

[764] Vgl. Kapitel 3.1.3.6.2.

[765] Vgl. Kapitel 3.1.3.6.1 und 3.1.3.6.2.

[766] Vgl. Kapitel 3.1.3.6.2.

von einem Service verstanden werden kann. Um dies zu erreichen, müssen Informationen unabhängig davon, in welchem Format diese einen Service erreichen, analysiert und in ein zur Verarbeitung angemessenes Format gebracht werden. Um einen dienstleistungsanbietenden Service erfolgreich ansprechen zu können, müssen vorher (1) dessen Adresse und (2) das verarbeitungsfähige Datenprotokoll sowie (3) die verarbeitungsfähigen Datenformate in Erfahrung gebracht werden. Um abschließend ein Dienstleistungsergebnis an den Dienstleistungsnachfrager zurückgeben zu können, muss entsprechend eine Nachricht an diesen vorbereitet und versendet werden. Diese Schritte zur Verarbeitung von Informationen müssen in einer SOA zusätzlich zur Dienstleistungserstellung durchgeführt werden. Vor allem auf Seiten der Kommunikationsinfrastruktur, bei der der Service-Bus, das Service-Registry und das Service-Repository diese Aufgaben in sehr hohem Maße auszuführen haben, kann dieser Aufwand sehr hoch werden. Weil Ressourcen und Zeit dadurch in Anspruch genommen werden, ergibt sich ein potentiell schlechtes Antwortzeitverhalten einzelner Services und der Kommunikationsinfrastruktur.[767]

Durch die Fähigkeit einer SOA, eine gute Skalierbarkeit zu erreichen,[768] kann das Antwortzeitverhalten der einzelnen Services leicht positiv beeinflusst werden, indem weitere Instanzen eines Services zur Dienstleistungsverarbeitung hinzugezogen werden. Demgegenüber stehen allerdings ein hohes Kommunikationsaufkommen und die damit verbundene Ressourcennutzung. Weitere Instanzen eines Services benötigen zusätzliche Ressourcen zur Kommunikation mit der Kommunikationsinfrastruktur, z. B. damit diese eine Lastverteilung zwischen existierenden Instanzen eines Services vornehmen. Auch muss jede Instanz – zumindest für ihre Lebenszeit – im Repository verwaltet werden. Dadurch wird die Möglichkeit eingeschränkt, ein besseres Antwortzeitverhalten durch Skalierung zu erreichen.

Aus diesen Gründen wird das Antwortzeitverhalten negativ beurteilt.

---

[767] Adam und McKendrick haben dies beispielhaft in einer Umfrage belegt: Die Effizienz ging aufgrund einer rapiden ansteigenden Verarbeitungsmenge an XML-Dokumenten zurück. Jedoch stuften nur 4 % der betroffenen Firmen dieses Performanzproblem als ernst ein. Es wird auch berichtet, dass Betroffene von Performanzproblemen diese Probleme mittels technischer Aufrüstung bzw. technischer Werkzeuge verringern können. Hier gibt es vielfältige Möglichkeiten, beispielsweise wurde Performanzverbesserung dadurch erreicht, dass Prozessoren aufgerüstet, die Bandbreite des Netzwerks erhöht, Hardwarekomponenten, die auf die Verarbeitung von XML spezialisiert sind, eingesetzt, oder Betriebssysteme aufgerüstet wurden. Vgl. Adam, McKendrick /Everything in Between/ S. 24-26.

Grad der Ressourcenausnutzung

Der Grad der Ressourcenausnutzung beschreibt die Fähigkeit einer Software, eine möglichst geringe Menge von Ressourcen zu benutzen, um Funktionen unter festgelegten Bedingungen auszuführen.[769]

Der Grad der Ressourcenausnutzung zur Dienstleistungsausführung ist in einer SOA dadurch hoch, weil zur Inanspruchnahme der Dienstleistung eines Services Kommunikation erforderlich ist. Um die notwendige Kommunikation durchführen zu können, ist zum einen eine Kommunikationsinfrastruktur und zum anderen, wie schon dargestellt, die Verarbeitung von Informationen notwendig. Eine Kommunikationsinfrastruktur, die für die gesamte Unternehmens-IT geeignet ist, benötigt Ressourcen. Besondere Anforderungen an die Kommunikationsinfrastruktur werden z. B. durch die örtliche Verteilung von Unternehmenssitzen gestellt. Auch die Verarbeitung von Informationen ist auf zusätzliche Ressourcen der Unternehmens-IT angewiesen.

Der hohe Bedarf an Ressourcen führt zu einer negativen Bewertung des Grades der Ressourcenausnutzung.[770]

Gehorsamkeit

Gehorsamkeit in Bezug auf Effizienz ist die Fähigkeit einer Software, Standards oder Vereinbarungen mit Bezug zur Effizienz einhalten zu können.[771]

Dienstleistungen bzw. die Ausführung genau abgegrenzter Funktionen werden in einer SOA von Services erbracht.[772] Dadurch ist die Überwachung einzelner Services mit Parametern wie Antwortzeitverhalten und Datendurchsatz möglich. Die Überwachung einzelner Services kann sicherstellen, dass Standards oder Vereinbarungen mit Bezug zur Effizienz eingehalten werden können. Wenn Vereinbarungen auf einer gröberen Ebene als der Ebene einzelner Services getroffen wurden, kann durch das Vorliegen von

---

[768] Vgl. Kapitel 3.2.2.2.1.

[769] Vgl. Kapitel 3.1.3.6.2.

[770] Der genaue Grad der Ressourcenausnutzung kann im Rahmen dieser Arbeit nicht beurteilt werden, da der Grad der Beanspruchung einen bestimmten Anteil der existierenden Ressourcen bedeutet. Diese Arbeit stellt eine Bewertung jedoch unabhängig von einer konkreten Ausprägung dar. Aus diesem Grund kann nur davon ausgegangen werden, dass sich aufgrund der aufgeführten Ausprägungen, eine höhere Auslastung der Ressourcen ergibt.

[771] Vgl. Kapitel 3.1.3.6.2.

Daten zu einzelnen Services nachverfolgt werden, welcher Service oder welche Services für Abweichungen verantwortlich sind. Folglich ist es Software-Entwicklern möglich, durch gezielte Maßnahmen für die Einhaltung von Standards oder Vereinbarungen zu sorgen.

Die Gehorsamkeit wird positiv bewertet.

3.2.2.5.2 Nutzenpotential der Effizienz

Effizienz beschreibt das Verhältnis zwischen einem erreichten Ergebnis und den dazu eingesetzten Ressourcen.[773] Der Ressourceneinsatz bezüglich Maschineneinsatz und Arbeitszeit wurde im Kapitel 3.2.2.5.1 bewertet. Anforderungen die erfüllt werden müssen stellen erreichte Ergebnisse dar und sind festgelegt.[774] Weitere, über Kapitel 3.2.2.5.1 hinausgehende und relevante Bewertungskriterien konnten nicht gefunden werden.

3.2.2.5.3 Gesamtbewertung der Effizienz

Nach McCall ergibt sich für die Effizienz eine negative Korrelation mit anderen Qualitätsattributen.[775] Beispielsweise geht McCall davon aus, dass bei Erreichungsgraden hoher Portabilität und Wandlungsfähigkeit[776] gleichzeitig die Effizienz gering ist. Die Sichtweise von McCall wird in dieser Arbeit bestätigt. Die Wandlungsfähigkeit und auch die Portabilität werden positiv bewertet. Zusammenfassend wird das Qualitätsattribut ‚Effizienz' negativ bewertet (Tab. 3-15).

---

[772] Vgl. Kapitel 2.2.4.1.

[773] Vgl. ISO /9000/ S. 15 (3.2.15).

[774] Vgl. Kapitel 3.1.3.6.2. Dies entspricht auch der Sichtweise des Minimumprinzips. Dieses besagt, dass eine vorgegebene Leistung mit möglichst geringem Mitteleinsatz erreicht werden soll. Gemäß Maximumprinzip ist bei gegebenem Mitteleinsatz eine möglichst hohe Leistung zu erzielen. Vgl. Antweiler /Wirtschaftlichkeitsanalyse/ S. 56.

[775] Vgl. McCall /Quality factors/ S. 967. McCall stellt in seiner Arbeit allerdings nur Behauptungen auf und argumentiert die von ihm dargestellten Zusammenhänge nicht.

[776] McCall bezeichnet ‚Wandlungsfähigkeit' im Sinne dieser Arbeit als ‚Wartbarkeit', vgl. Kapitel 3.1.3.3.

(Unter-)Qualitätsattribute	*Wirkung*
Antwortzeitverhalten	–
Grad der Ressourcenausnutzung	–
Gehorsamkeit	+
Gesamtbewertung	–

Tab. 3-15: Nutzenbewertung des Qualitätsattributes Effizienz

3.2.2.6 Wiederverwendbarkeit

Die Wiederverwendbarkeit ist die Fähigkeit einer Software bzw. Softwarekomponente, in mehr als einem Anwendungsfall Verwendung zu finden.

Im Folgenden werden auf Basis von Argumentationen Bewertungen der Unterqualitätsattribute der Wiederverwendbarkeit vorgenommen. Daran anschließend werden die Wirkungen einer SOA auf die Wiederverwendbarkeit einer Unternehmens-IT untersucht und es wird eine Gesamtbewertung aufgestellt.

3.2.2.6.1 Nutzenpotential der Unterqualitätsattribute der Wiederverwendbarkeit

Die Unterqualitätsattribute der Wiederverwendbarkeit sind Allgemeingültigkeit, Modularität und Selbstbeschreibung.[777]

Allgemeingültigkeit

Allgemeingültigkeit stellt das Ausmaß von Softwarekomponenten dar, und beschreibt, inwieweit der Funktionsumfang einzelner Softwarekomponenten geringe Spezialisierung aufweist.[778]

Die Allgemeingültigkeit von Softwarekomponenten kann auf zwei Ebenen betrachtet werden: zum einen auf einer technischen, zum andern auf einer funktionalen Ebene. Wie schon erwähnt kann eine SOA auf Basis unterschiedlicher technischer Lösungen

[777] Vgl. Kapitel 3.1.3.7.2.

[778] Vgl. Kapitel 3.1.3.7.2.

aufbauen. Die technischen Aspekte werden im Rahmen dieser Arbeit nicht näher untersucht.[779] Die Allgemeingültigkeit technischer Lösungsmöglichkeiten einer SOA hängt jedoch grundlegend von zwei Rahmenbedingungen ab. Erstens, ob Services die vorhandene Kommunikation (verwendete Formate, Schnittstellen und Protokolle) verstehen und unterstützen können; zweitens, ob Services in einer vorhandenen technischen Umgebung lauffähig (Plattformunterstützung) sind. Diese Fragen stellen sich jedenfalls bei jeder potentiellen Wiederverwendung von Software bzw. Softwarekomponenten. Der technische Aspekt der Allgemeingültigkeit wird neutral bewertet.

Betrachtet man die funktionale Ebene so kann festgestellt werden, dass Services abgegrenzte, an Geschäftsprozessen orientierte Funktionen abdecken.[780] Durch die Orientierung an Geschäftsprozessen sind Dienstleistungen derart in Services gekapselt, dass diese durch die Unternehmens-IT so genutzt werden, wie die Geschäftsprozesse diese benötigen.[781] Services sind ebendeshalb auf die Bedürfnisse der Geschäftsprozesse eines Unternehmens zugeschnitten. Allgemeingültig ist ein Service somit maximal für den Unternehmenskontext: eine Dienstleistung wird durch einen Service genau so erfüllt, dass die Erfüllung den Vorgaben und Interessen des Unternehmens genügt. Beispielsweise kann eine Dienstleistung unter Beachtung der Erhebung bestimmter Daten, die nur für das Unternehmen relevant sind, durchgeführt werden. Enthalten Geschäftsprozesse in einem Unternehmen gleichartige Prozessschritte, die bei der Analyse der Geschäftsprozesse erkannt wurden, ergibt sich ein Potential zur Wiederverwendung, weil Services den Anforderungen des Unternehmens genügen. Wiederverwendung in fremden Unternehmen ist jedoch schwieriger zu erreichen, weil davon ausgegangen werden kann, dass Services einen prinzipiell zu geringen Allgemeinheitsgrad erreichen. Die Verwendung von Services über Unternehmensgrenzen hinweg erscheint trotzdem möglich, weil externe Unternehmen, die mit der Unternehmens-IT direkt interagieren wollen, auch an entsprechenden Prozessen des Unternehmens beteiligt sind. Die Allge-

---

[779] Vgl. Kapitel 1.3.3.

[780] Vgl. Kapitel 2.2.4.1.

[781] Im Gegensatz zu beispielsweise objektorientierter Programmierung, in der Funktionen und Daten so gekapselt werden, dass diese Objekte der Umwelt des Unternehmens entsprechen, unabhängig davon, welche Funktionen und in welchem Zusammenhang diese benötigt werden.

meingültigkeit der Unternehmensebene ist deswegen für diese Art der Verwendung ausreichend allgemein.[782]

Auslöser für Weiterentwicklungen und Änderungen der Unternehmens-IT sind Erweiterungen und Änderungen der Struktur oder der Geschäftsprozesse eines Unternehmens. Änderungen von *Strukturen* haben Einfluss auf die Unternehmens-IT, weil hier z. B. Zugriffsberechtigungen oder der Ursprungsort der Dienstleistungsanforderung geändert wird. Dienstleistungen einer Unternehmens-IT müssen deswegen ggf. von Orten und Personen genutzt werden, von denen bisher keine Verwendung stattgefunden hat. Die Änderungen der *Geschäftsprozesse* dagegen bedingen mindestens eine Anpassung der Choreographie von Services; darüber hinaus ggf. auch die Änderung und Erweiterung der Unternehmens-IT, wodurch auch vorhandene Services betroffen seien können. Hier können nun zwei Ausprägungen des Funktionsumfangs von Services betrachtet werden:[783] Services mit hohem Funktionsumfang und Services mit niedrigem Funktionsumfang. Für den Fall der Implementierung von Services mit hohem Funktionsumfang, ist die Wahrscheinlichkeit hoch, dass Services geändert werden müssen. Im Fall der Implementierung von Services mit niedrigem Funktionsumfang, ist eine neue Orchestrierung mit höherer Wahrscheinlichkeit möglich, um zu einer angepassten Choreographie zu gelangen als bei der Implementierung von Services mit hohem Funktionsumfang. Dies liegt daran, weil Services mit hohem Funktionsumfang einen höheren Teil eines Geschäftsprozesses abdecken als Services mit niedrigem Funktionsumfang. Somit sind Services mit hohem Funktionsumfang bei Änderungen eines Geschäftsprozesses mit höherer Wahrscheinlichkeit betroffen. Häufig umfassen die Änderungen von Geschäftsprozessen neue, geänderte oder neu angeordnete Prozessschritte. Services mit niedrigerem Funktionsumfang können dem neuen Geschäftsprozess durch Einfügen neuer Services, durch Änderung einzelner Services geringen Funktionsumfangs oder durch eine neue Choreographie der Services gerecht werden. Bezüglich dieses Kriteriums ist die Allgemeingültigkeit somit abhängig von der Ausprägung des Funktionsumfangs der

---

[782] In der Argumentation wurde die (Wieder-)Verwendung von Services angesprochen, indem Instanzen über die Kommunikationsinfrastruktur – aus dem Unternehmen oder von außerhalb kommend – verwendet werden. Die Wiederverwendung der Implementierung in einer anderen Unternehmens-IT, wird hier nicht betrachtet, weil davon ausgegangen wird, dass ein Unternehmen für die eigene Unternehmens-IT implementiert und nicht für den Markt als Software-Entwickler tätig ist. Diese Perspektive kann dennoch relevant sein. In diesem Fall steht das Unternehmen allerdings grundsätzlich vor der Herausforderung, Funktionen derart Allgemeingültig zu gestalten, dass diese als Standardprodukte angeboten werden können.

Services. Dieser unterliegt jedoch den Gestaltern einer Unternehmens-IT und wird nicht pauschal vorgegeben.[784] Die Argumentation zeigt jedoch, dass ein niedriger Funktionsumfang bezüglich der Allgemeinheit tendenziell geeigneter erscheint als ein hoher. Eine Unterstützung zur Bewertung liefert dieses Kriterium jedoch nicht.

Wird ein neuer Geschäftsprozess, z. B. die Verwaltung oder Vermarktung neuer Produkte, in der Unternehmens-IT implementiert und existieren Geschäftsprozesse, die ähnliche Aufgaben durchführen, können existierende Services mit entsprechenden Funktionen wieder genutzt werden. Die Wiederverwendung gestaltet sich dabei durch die lose Kopplung sowie die Kommunikationsmöglichkeit über Schnittstellen und die Kommunikationsinfrastruktur – und zwar, weil bei der Erstellung der Choreographie nur bestehende Funktionen über bekannte Mechanismen eingebunden werden müssen.

Die Allgemeingültigkeit zur Wiederverwendung wird unternehmensintern positiv beurteilt. Unternehmensübergreifend zur unabhängigen Verwendung durch fremde Unternehmen in ihrer Unternehmens-IT sind Services aufgrund der Ausrichtung der Dienstleistung am Unternehmen nicht geeignet. Jedoch ist eine Einbeziehung anderer Unternehmen zur Automatisierung von Geschäftsprozessen möglich. Ebenso kann davon ausgegangen werden, dass Services, die in einem Unternehmen zur Nutzung in diesem Unternehmen implementiert wurden, nicht als eigenständiges Produkt anderen Unternehmen angeboten werden sollen. Die Allgemeingültigkeit wird positiv bewertet.

Modularität

Modularität stellt das Ausmaß einer Software dar, inwieweit die einzelnen Softwarekomponenten voneinander unabhängig strukturiert sind.[785]

Eine Wiederverwendung vorhandener Softwarekomponenten wird dadurch unterstützt, dass spezifische Funktionen, die nachgefragt werden, von einer Softwarekomponente bereitgestellt werden können. SOA sind modular aufgebaut. Dies erkennt man daran, dass Services durch ihre lose Kopplung zu anderen Services und durch eine ausschließliche Kommunikation über die Kommunikationsinfrastruktur unabhängig voneinander

---

[783] Vgl. Kapitel 2.2.4.1.

[784] Vgl. Kapitel 2.2.4.1.

[785] Vgl. Kapitel 3.1.3.7.2.

strukturiert sind.[786] Weil sich der Funktionsumfang von Services an Geschäftsprozessen orientiert, sind Funktionen schon so in Module (die Services) aufgeteilt, wie diese von der Unternehmens-IT nachgefragt werden könnten. Die Modularität einer SOA wird deswegen positiv bewertet.

Selbstbeschreibung

Selbstbeschreibung stellt die Fähigkeit einer Software dar, Erläuterungen einer Funktion zur Verfügung zu stellen.[787]

Durch das Vorhandensein eines Service-Repositorys als Bestandteil der notwendigen Kommunikationsinfrastruktur einer SOA sind Informationen zu jedem Service an zentraler, von allen Services erreichbarer Stelle gespeichert und abrufbar. Diese Informationen werden entweder selbstständig von Services an das Service-Repository geliefert, sobald diese in einer Unternehmens-IT zur Ausführung gebracht werden. Oder sie werden durch den Software-Entwickler vor Ausführung eines Services an das Service-Repository gegeben. Im ersten Fall werden die Informationen des Service-Repository darüber hinaus mindestens noch in den einzelnen Services gespeichert und jeder Service beinhaltet die Funktion, die entsprechenden Informationen weitergeben zu können.

Informationen bezüglich des Serviceaufrufs, d. h. welche Schnittstellen mit welcher Funktionalität existieren, werden im Service-Repository gespeichert. Die Argumentation erfolgt analog zu der entsprechenden Argumentation der Informationen zu Services.

Wird in einer SOA neben einem Service-Repository noch ein Service-Registry verwendet, findet die Selbstbeschreibung weitere Unterstützung. Ein Service-Registry ermöglicht, dass die Informationssuche auf einer semantischen Ebene erfolgt. Wenn ein Service gesucht wird, der Name oder andere eindeutige Identifizierer aber unbekannt sind, ermöglicht ein Service-Repository durch Analyse semantischer Anfragen das Finden des potentiell gesuchten Services. Durch ein Service-Registry steigt somit die Selbstbe-

[786] Vgl. Kapitel 2.2.4.3.

[787] Vgl. Kapitel 3.1.3.7.2.

schreibung der gesamten Unternehmens-IT, weil semantische Auskunftswünsche bearbeitet werden können.[788]

Die strukturelle Beschreibung der Unternehmens-IT ist in einer SOA potentiell gut ableitbar. Weil Services durch Choreographien so zusammengeschaltet werden, dass Geschäftsprozesse erfolgreich bearbeitet werden können, Services jedoch unabhängig voneinander existieren, müssen Informationen zur Choreographie der Services an mindestens einer Stelle im Unternehmen gespeichert sein.[789] Diese Informationen liegen Software-Entwicklern folglich vor und können beispielsweise graphisch dargestellt werden dargestellt.

Zusammenfassend wird die Selbstbeschreibung einer SOA positiv bewertet.

3.2.2.6.2 Nutzenpotential der Wiederverwendbarkeit

Wie in Kapitel 3.1.3.7 beschrieben können zwei Ebenen der Wiederverwendung unterschieden werden. Zum einen Wiederverwendung von Softwarekomponenten in anderen Unternehmen. In diesem Fall werden Softwarekomponenten mit dem Ziel entwickelt, diese einem Nachfragermarkt zugänglich machen zu können. Zum anderen Wiederverwendung von Softwarekomponenten im eigenen Unternehmen bzw. durch Partner eines Unternehmens, ohne die Kontrolle über die jeweiligen Softwarekomponenten abzugeben.

Durch die Möglichkeit, eine komplette Unternehmens-IT auf Basis einer SOA aufbauen zu können, können Services unternehmensweit (und sogar über die Grenzen des Unternehmens hinaus) angeboten werden. Durch eine einheitliche Kommunikationsinfrastruktur können Services in Unternehmen auf bekannte Art und Weise verwendet werden. Die Fähigkeit einer SOA, ihre Dienste unternehmensweit anbieten zu können, erhöht die Wahrscheinlichkeit, dass Funktionen, die an anderer Stelle im Unternehmen benötigt werden, innerhalb weiterer Geschäftsprozesse genutzt werden. Dies begründet sich dadurch, weil die Anzahl potentieller Diensleistungsnachfrager höher ist als wenn

---

[788] Beispielsweise ist vorstellbar, dass eine Anfrage eines Software-Entwicklers „Suche alle Services, die Kundendaten verarbeiten“ durch ein Service-Registry erfolgreich und mit dem gewünschten Ergebnis bearbeitet werden kann.

[789] In der Praxis werden hier Techniken wie BPEL und vergleichbare diskutiert. Vgl. Linthicum /Next Generation/ S. 279-280; Adam, McKendrick /Everything in Between/ S. 16; Linthicum /Next Generation/ S. 291; vgl. weiterführend Pasley /BPEL/.

ein Service nur einer eingegrenzten Anzahl potentieller Nutzer zugänglich gemacht wird. Der Fokus des Konzepts einer SOA erhöht somit die Wahrscheinlichkeit, Wiederverwendung im Unternehmen realisieren zu können.

Wiederverwendung in fremden Unternehmen ist mit Services einer SOA schwer möglich. Funktionen eines Services sind so zugeschnitten, dass ein Unternehmen diese als Dienstleistung mit genau festgelegten Rahmenbedingungen verwenden kann. Zu solchen individuell festgelegten Rahmenbedingungen gehören z. B. Verarbeitungsreihenfolgen, Datenformate und Dateninhalte. Zwar könnte versucht werden, Services derart allgemeingültig zu gestalten, dass bestimmte grob beschriebene Funktionen unter verschiedenen Rahmenbedingungen einsatzfähig sind. In einer Unternehmens-IT wird eine solche Implementierung jedoch nicht zu rechtfertigen sein. Der dazu notwendige Aufwand, einen Service auf technischer und fachlicher Ebene so allgemeingültig zu halten, dass diese ohne Überarbeitungsaufwand in anderen Unternehmen eingesetzt werden kann, übersteigt den potentiellen Gewinn, der aus Einsätzen außerhalb eines Unternehmens erzielt werden könnte.[790]

Die Wiederverwendung eines Services findet nur statt, wenn ein Service von unterschiedlichen Stellen ausgenutzt werden kann. Rine, Nada und Jaber haben gezeigt, dass die Nutzung von Adaptern in einem komponentenorientierten Softwaresystem die Wiederverwendbarkeit einzelner Softwarekomponenten erhöht.[791] Im Verständnis von Rine, Nada und Jaber wird ein Adapter dazu genutzt, die Dienstleistung der angebundenen Softwarekomponente anderen Softwarekomponenten zugänglich zu machen. Diese Definition entspricht der hier getroffenen Definition der Serviceschnittstellen.[792] Deswegen können die Ergebnisse der Studie von Rine, Nada und Jaber auf eine SOA übertragen werden: Das Vorhandensein von u. U. mehreren Schnittstellen zu einem Service erhöht das Potential, dass der entsprechende Service wiederverwendet wird.

---

[790] Als Indiz dafür wird auch der Fakt gezählt, dass Unternehmen, die nicht in der Softwarebranche sind, ihre intern entwickelten Softwaresysteme nicht anderen Unternehmen zum Kauf anbieten.

[791] Vgl. Rine, Nada, Jaber /Adapters/ S. 38. Im weiteren Verlauf ihres Artikels beschreiben Rine, Nada und Jaber allerdings, dass eine Verwendung über ‚Adapter' einfach möglich ist. Adapter im Verständnis der Autoren sind in dieser Arbeit Serviceschnittstellen. Auf diesen Aspekt der Wiederverwendung wird in Kapitel 3.2.2.7.2 eingegangen.

[792] Vgl. Kapitel 2.2.4.2.

3.2.2.6.3 Gesamtbewertung der Wiederverwendbarkeit

Zusammenfassend wird das Qualitätsattribut ‚Wiederverwendbarkeit' positiv bewertet (Tab. 3-16).

(Unter-)Qualitätsattribute	*Wirkung*
Hohe Anzahl potentieller Dienstleistungsnachfrager	+
Unternehmens-externe Wiederverwendung	–
Gute Möglichkeit zur Einbindung von Services	+
Allgemeingültigkeit	+
Modularität	+
Selbstbeschreibung	+
Gesamtbewertung	+

Tab. 3-16: Nutzenbewertung des Qualitätsattributes Wiederverwendbarkeit

3.2.2.7 Portabilität

Portabilität ist die Fähigkeit einer Software, von einer Umgebung in eine andere übertragen werden zu können.[793] Dabei kann die Umgebung unterschiedlich sein in Bereichen der Soft- und Hardware wie auch in organisatorischen Bereichen.

Im Folgenden werden auf Basis von Argumentationen Bewertungen der Unterqualitätsattribute der Portabilität vorgenommen. Daran anschließend werden die Wirkungen einer SOA auf die Portabilität einer Unternehmens-IT untersucht und es wird eine Gesamtbewertung aufgestellt.

3.2.2.7.1 Nutzenpotential der Unterqualitätsattribute der Portabilität

Die Unterqualitätsattribute der Portabilität sind Anpassungs- und Installationsfähigkeit sowie Koexistenz und Ersetzbarkeit.[794]

---

[793] Vgl. Kapitel 3.1.3.8.2.

Anpassungsfähigkeit

Anpassungsfähigkeit ist die Fähigkeit einer Software, in unterschiedlichen (spezifizierten) Umgebungen lauffähig zu sein, ohne dass Aktivitäten durchgeführt werden müssen, die nicht explizit zur Anpassung vorgesehen sind.[795]

Damit ein Service in einer anderen Umgebung als der, in der er bisher eingesetzt wurde, lauffähig ist, muss zum einen eine technische Lauffähigkeit vorhanden sein (dieser Aspekt der Installationsfähigkeit wird weiter unten bewertet); zum anderen muss der Service so anpassbar sein, dass wenn der Service angesprochen wird, seine Dienstleistung initiiert und die Arbeitsergebnisse zurückgeliefert werden. Eine solche Anpassbarkeit kann über zwei Wege erreicht werden: erstens kann die Kommunikationsinfrastruktur auf die neue Umgebung erweitert werden, zweitens können die Schnittstellen des Services so angepasst werden, dass sie die Kommunikation bis zur entfernten Kommunikationsinfrastruktur unterstützen. Eine solche Erweiterung stellt die Voraussetzung zur Nutzung eines Services dar.

Die Anpassungsfähigkeit von Softwarekomponenten einer SOA wird negativ bewertet.

Installationsfähigkeit

Installationsfähigkeit bezeichnet die Fähigkeit einer Software, in unterschiedlichen Umgebungen installiert werden zu können.[796]

Die Installationsfähigkeit ist, wie oben erwähnt, eine Grundvoraussetzung für einen Service, um in einer neuen Umgebung lauffähig sein zu können. Die Installationsfähigkeit ist sehr stark von der jeweiligen eingesetzten Technik abhängig, sowohl auf Seiten der neuen als auch der alten Umgebung.[797] Deswegen wird hier auch häufig von der Plattformunabhängigkeit[798] einer Software gesprochen. Wie in Kapitel 1.3.3 dargestellt, erfolgt eine Bewertung möglicher Technologien – und für diesen Fall die damit verbun-

---

[794] Vgl. Kapitel 3.1.3.8.2.

[795] Vgl. Kapitel 3.1.3.8.2.

[796] Vgl. Kapitel 3.1.3.8.2.

[797] Beispielsweise ist ein Aspekt, dass die Programmiersprache, in der der Service implementiert wurde, einen Quellcode liefert, der auf der entsprechenden neuen Plattform lauffähig ist.

[798] Die Plattform bedeutet hierbei die technische Grundlage (sowohl Hardware als auch Software, wie z. B. Betriebssysteme) auf der eine Software abläuft. Plattformunabhängigkeit bedeutet, dass eine Software auf unterschiedlichen Plattformen ablaufen kann, ohne angepasst zu werden.

dene Installationsfähigkeit – in dieser Arbeit nicht. Installationsfähigkeit kann generell auf zwei Wegen erreicht werden. Zum einen durch die Verwendung von Quellcode-Generatoren[799], die Services für unterschiedliche Plattformen aus einer Quelle stammend erzeugen können, zum anderen durch die Verwendung von Technologien, die auf unterschiedlichen Plattformen zum Ablaufen gebracht werden können.[800] Eine SOA legt jedoch keine Hilfen oder Gestaltungsrichtlinien fest, die die Verwendung von Quellcode-Generatoren oder von plattformunabhängigen Technologien vorschreibt.[801] Die Verwendung von Quellcode-Generatoren oder plattformunabhängigen Technologien wird freilich auch nicht verboten oder verhindert. Deswegen kann eine Bewertung der Installationsfähigkeit nicht hergeleitet werden.

Ersetzbarkeit

Ersetzbarkeit ist die Fähigkeit einer Software, eine andere Software, welche für gleiche Zwecke eingesetzt wurde, in der entsprechenden Umgebung ersetzen zu können.[802]

Unter der Voraussetzung, dass eine Anpassung an die neue Umgebung erreicht werden kann, ist eine SOA durchaus verwendbar, um bisherige Funktionen durch Services anbieten zu können, indem die gesamte bestehende Software abgelöst wird. Müssen jedoch verschiedene Systeme, ein System auf Basis einer SOA und ein System, das nicht auf Basis einer SOA aufbaut, parallel betrieben werden, muss spezielle Software hinzugefügt werden, die die Kommunikation zwischen den Systemen durchführen kann. Selbst dann kann es sein, dass die Struktur des ‚alten' Systems nicht geeignet ist, Services einer SOA zu verwenden. Eine teilweise Ersetzbarkeit ist unter Umständen folglich nur durch zusätzlichen Aufwand erreichbar, während eine vollständige Ersetzbarkeit

---

799 Banke, Krafzig und Slama beschreiben den Ansatz einer Model-Drive-Architecture (MDA). Der Ansatz einer MDA verfolgt die Idee, aus einem plattformunabhängigen Modell einer Software (oder einer Softwarekomponente) automatisch einen ablauffähigen Quellcode für unterschiedliche Plattformen erstellen zu können. Vgl. Banke, Krafzig, Slama /Enterprise SOA/ S. 167-168.

800 Hierunter fallen z. B. sog. Bytecode-Technolgogien wie Java, die eine ‚Virtual Machine' zum Ablaufen benötigen, diese jedoch existiert für viele Plattformen, weshalb ein Service auf Basis von Bytecode-Technologien auf unterschiedlichen Plattformen ablauffähig ist, oder die Verwendung von Applikationsstandards wie Web Services, für die Unterstützung für unterschiedliche Plattformen vorhanden sind. Vgl. Chappell /ESB/ S. 3-4.

801 Vgl. Natis /SOA/ S. 25.

802 Vgl. Kapitel 3.1.3.8.2.

keinen zusätzlichen Aufwand erfordert.[803] Durch die negative und neutrale Ausprägung wird die Ersetzbarkeit als Unterqualitätsattribut der Portabilität insgesamt als negativ bewertet. Ergänzt werden soll diese Einschätzung noch um den Hinweis, dass auch diese Bewertung in der Praxis stark von den jeweiligen eingesetzten Technologien abhängt, dieser Aspekt aber – wie mehrfach erwähnt – nicht bewertet wird.[804]

3.2.2.7.2 Nutzenpotential der Portabilität

Portabilität in Bereichen der Soft- und Hardware wird in dieser Arbeit nicht explizit bewertet, da von spezifischen Technologien abstrahiert wird.[805] Die Bewertung der Portabilität müsste vor allem in diesen Bereichen unter Beachtung vielfältiger Kombinationsmöglichkeiten vorgenommen werden. Dies würde den Umfang dieser Arbeit jedoch übersteigen. Es wird jedoch angenommen, dass eine Portabilität zwischen verschiedenen Technologien tendenziell eher mit Problemen behaftet ist als dass eine reibungslose Portabilität vorgefunden werden kann. Deswegen wird die technische Portabilität negativ bewertet.

Portabilität nach organisatorischen Kriterien kann nach Anforderungen der Ablauf- und Aufbauorganisation differenziert werden.[806] Die Portabilität einer SOA in unterschiedliche *Ablauforganisationen* wird deswegen positiv bewertet, da Services durch die Bildung neuer Choreographien schnell in neue organisatorische Abläufe überführt werden können. Beispielsweise kann der Prozess der Auftragsannahme durch einen mobilen Vertriebsmitarbeiter, der beim Kunden vor Ort ist, anders strukturiert sein als der Prozess der Auftragsannahme auf einer Website durch den Kunden selber, weil der Kunde offline direkt bar zahlen kann und keiner Bonitätsprüfung unterzogen werden muss. Die Portabilität einer Software (bzw. ihrer Softwarekomponenten) in eine unterschiedliche *Aufbauorganisation* wird ebenso positiv bewertet. Die Änderung einer Aufbauorganisation kann die Notwendigkeit mit sich bringen, dass Geschäftsprozesse an einem anderen

---

[803] Der Aufwand, ein gesamtes ‚Altsystem' (sog. ‚Legacy system') umstellen zu müssen, wird nicht betrachtet, weil hier die generelle Ersetzbarkeit geprüft wird.

[804] Vgl. Kapitel 1.3.3.

[805] Vgl. Kapitel 1.3.3.

[806] Vgl. zur Aufbau- und Ablauforganisation Schreyögg /Organisation/ S. 119-120; Mellis /Projektmanagement/ S. 72.

Ort ausgeführt werden müssen als bisher. Services schränken die örtliche Einsetzbarkeit nicht ein.[807]

Nach Raasch hängt die Portabilität eines Softwaresystems entscheidend von der eingesetzten Softwarearchitektur ab.[808] Raasch zeigt, dass Portabilität besonders dann gut erreichbar ist, wenn eindeutig definierte und langlebige Schnittstellen existieren. In einer SOA liegt beides vor. Schnittstellen müssen für jeden Service implementiert werden, damit die Dienstleistung des Services genutzt werden kann. Schnittstellen können durch Versionierung auch eine sehr lange Lebensdauer erreichen, weil sie bei Weiterentwicklungen eines Services nicht ausgetauscht werden müssen, sondern solange weiterverwendet werden können, bis alle Dienstleistungsnehmer die weiterentwickelten Schnittstellen benutzen. Der Argumentation von Raasch folgend ist eine Software auf Basis einer SOA portabel.

Portabilität einer Software umfasst auch die Möglichkeit, eine Software auf portablen Geräten verwenden zu können. Neben den weiter unten bewerteten Aspekten der Anpassungs- und Installationsfähigkeit auf portablen Geräten kann in dieser Hinsicht noch nach der *Echtzeitverfügbarkeit* der Softwarekomponenten und der Daten unterschieden werden. Sollen Softwarekomponenten und Daten in Echtzeit verfügbar sein, so muss von portablen Geräten eine Verbindung zu den Softwarekomponenten bestehen, über die auf Services und Daten zugegriffen werden kann. Zu diesem Zweck müssen Informationen zwischen dem portablen Gerät und denjenigen Softwarekomponenten ausgetauscht werden, die Zugriff auf aktuelle Daten haben. Diese Softwarekomponenten sind diejenigen, die in der nicht-portablen Umgebung einer Unternehmens-IT eingesetzt werden. In einer SOA muss deshalb eine Kommunikation zwischen nicht-portabel eingesetzten und portabel eingesetzten Services bzw. Teilen der Infrastrukturplattform stattfinden. Diese Kommunikation läuft über die Kommunikationsinfrastruktur. Wie im Kapitel 3.1.3.4.1 gezeigt, ist der zur Kommunikation notwendige Datenumfang zwischen Services hoch ausgeprägt und das Bruttovolumen kann das Nettovolumen weit übertreffen. Dieses Datenvolumen schränkt die Einsatzmöglichkeit auf portablen Geräten ein, weil eine hohe Datenrate zur Übermittlung der Datenmenge zwischen portablem Gerät und der nicht-portablen Unternehmens-IT notwendig ist, um das Nettovolumen

---

[807] Vgl. McGovern u. a. /Architecture/ S. 59.

[808] Vgl. Raasch /Systementwicklung/ S. 34-35.

übermitteln zu können. Es kann jedoch davon ausgegangen werden, dass Technologien zur Verfügung stehen werden, die entsprechende Datenraten realisieren können und eine Nutzung von portablen Geräten ohne Einschränkung erlauben.[809] Soll keine Echtzeitverfügbarkeit vorliegen, sind die Probleme des Bruttovolumens einer SOA nicht gegeben. Jedoch können sich Probleme aus redundanter Datenhaltung[810] und aus der Notwendigkeit der Synchronisierung[811] der eingesetzten Services auf einem portablen Gerät ergeben. Aus diesen Gründen ist die Bewertung der Echtzeitverfügbarkeit für portable Geräte neutral.

An dieser Stelle sei darauf hingewiesen, dass die Portabilität die Fähigkeit einer Software darstellt, von einer Umgebung in eine andere übertragen werden zu können.[812] Im Zusammenhang mit global bzw. verteilt agierenden Unternehmen kann jedoch auch die *Portabilität von Funktionen* relevant sein. Dies bedeutet, dass nicht eine gesamte Software oder Teile einer Software portabel sind, sondern, dass angebotene Funktionen einer Software oder Teile einer Software portabel genutzt werden können. Die Software verrichtet die Arbeit in ihrem gewohnten Umfeld und stellt Funktionen zur Nutzung bereit. Services – genauer: die Serviceschnittstellen – einer SOA können über die Kommunikationsinfrastruktur[813] gefunden und aufgerufen werden.[814] Voraussetzung dafür ist, dass die eingesetzte Kommunikationsinfrastruktur in der Lage ist, eine verteilte Unternehmens-IT zu unterstützen.[815] Dadurch ist es unerheblich, von welchem Ort ein Dienstleistungsnehmer Leistungen in Anspruch nimmt. Deswegen ermöglicht eine

---

[809] Vgl. Kapitel 3.1.3.6.3.

[810] Vgl. zu Problemen redundanter Datenhaltung Raasch /Systementwicklung/ S. 133-134; zu Redundanzen im Allgemeinen Ferstl, Sinz /Wirtschaftsinformatik/ S. 198-206.

[811] Werden Teile einer Unternehmens-IT auf portablen Geräten eingesetzt und sind diese nicht so konfiguriert, dass die Softwarekomponenten der nicht-portablen Unternehmens-IT in Echtzeit verwendet werden, müssen Teile der Unternehmens-IT auf entsprechenden portablen Geräten installiert werden. Diese Teile der Unternehmens-IT müssen sich den Änderungen und Erweiterungen der Unternehmens-IT entsprechend anpassen, damit einheitliche Prozesse und Datenstrukturen in einer Unternehmens-IT verwendet werden.

[812] Vgl. Kapitel 3.1.3.8.2.

[813] Vgl. Kapitel 2.2.4.3.

[814] Vgl. Rine, Nada, Jaber /Adapters/ S. 38-40. Rine, Nada und Jaber verwenden in ihrer Arbeit den Begriff ‚Adapter'. Adapter sind nach dem Begriffsverständnis dieser Arbeit Serviceschnittstellen (vgl. S. 39). Rine, Nada und Jaber beschreiben, dass eine Wiederverwendung von Softwarekomponenten durch die Nutzung von ‚Adaptern' (=Serviceschnittstellen) auch über verschiedene Umgebungen hinweg möglich wird.

SOA eine portable und verteilte Verwendung von Funktionen einer Unternehmens-IT.[816]

3.2.2.7.3 Gesamtbewertung der Portabilität

Zusammenfassend wird das Qualitätsattribut ‚Portabilität' neutral bewertet (Tab. 3-17).

(Unter-)Qualitätsattribute	*Wirkung*
Soft- und Hardwareportabilität	% (–)[817]
Organisatorische Unterstützung	+
Schnittstellenunterstützung	+
Echtzeitverfügbarkeit	0
Portabilität von Funktion	+
Anpassungsfähigkeit	–
Installationsfähigkeit	%[818]
Ersetzbarkeit	–
Gesamtbewertung	%

Tab. 3-17: Nutzenbewertung des Qualitätsattributes Portabilität

3.2.3 Nutzenpotentiale aus geschäftlicher Sicht

Im Folgenden wird das Konzept einer SOA anhand der geschäftlichen Qualitätsattribute des in Kapitel 3.1.4 aufgestellten Teils des Qualitätsmodells bewertet.

---

[815] Dies jedoch ist eine technische Frage und wird an dieser Stelle nicht weiter untersucht. Vgl. Kapitel 1.3.3. Es sei darauf hingewiesen, dass heute verfügbare Kommunikationsplattformen wie NetWeaver, WebSphere u. s. w. dazu in der Lage sind. Vgl. z. B. für NetWeaver Karch u. a. /SAP/ S. 33-36.

[816] Vgl. ergänzend zur Unterstützung verteilter Systeme durch SOA Kapitel 3.2.3.3.1.

[817] Siehe Text (Kapitel 3.2.2.7.2).

[818] Siehe Text (Kapitel 3.2.2.7.1).

3.2.3.1 Strategieunterstützung

Strategieunterstützung bedeutet die Fähigkeit einer Software, dem Unternehmensgestalter die Freiheit zu geben, Unternehmensstrategien formulieren und verfolgen zu können, ohne dass Einschränkungen durch die Software bestehen.[819]

Im Folgenden werden auf Basis von Argumentationen Bewertungen der Unterqualitätsattribute der Strategieunterstützung vorgenommen. Anschließend werden die Wirkungen einer SOA auf die Strategieunterstützung einer Unternehmens-IT untersucht und es wird eine Gesamtbewertung aufgestellt.

3.2.3.1.1 Nutzen der Unterqualitätsattribute der Strategieunterstützung

Die Unterqualitätsattribute der Strategieunterstützung sind ‚Unterstützung von Gesamtunternehmensstrategien', ‚Unterstützung von Wettbewerbsstrategien' und ‚Unterstützung des Strategiewandels'.[820]

Unterstützung von Gesamtunternehmensstrategien

Unterstützung von Gesamtunternehmensstrategien ist die Fähigkeit einer Softwarearchitektur, Produkt-Markt-Strategien unterstützen zu können.

Produkt-Markt-Strategien legen erstens fest, in welche Produkt- bzw. Dienstleistungsbereiche verstärkt investiert werden soll und welche eher restriktiv oder auch abschöpfend behandelt werden sollen;[821] zweitens bestimmen sie, auf welchen Märkten die Produkte oder Dienstleistungen angeboten werden sollen.

Eine Softwarearchitektur hat keinen Einfluss auf die Fähigkeit, bestimmte Produkte oder Dienstleistungen anzubieten. Der Grund dafür ist, dass das Anbieten eines beliebigen Produktes oder einer beliebigen Dienstleistung durch jede Software unterstützt werden kann. Demnach hat auch die jeweilige Softwarearchitektur keinen Einfluss auf die strategische Entscheidung, in welche Produkt- bzw. Dienstleistungsbereiche investiert werden soll.

---

[819] Vgl. Kapitel 3.1.4.2.2.

[820] Vgl. Kapitel 3.1.4.2.2.

[821] Vgl. zum folgenden Absatz Macharzina /Unternehmensführung/ S. 203.

Eine Software und folglich auch die ihr zugrunde liegende Softwarearchitektur hat aber durchaus Einfluss auf die potentiellen Nachfragermärkte.[822] Der Einfluss macht sich durch die Möglichkeit, Teilnehmer entsprechender Märkte über eine Software ansprechen und bedienen zu können bemerkbar. Eine Software auf Basis einer SOA bietet die Möglichkeit, Kunden (und auch Lieferanten und weiteren Geschäftspartnern) bestimmte Dienstleistungen zu Produkten bzw. direkt entsprechende Dienstleistungen anbieten zu können, indem Schnittstellen zur Unternehmens-IT bereitgestellt werden.[823] Auf diese Weise können neue Märkte mit zusätzlichen – durch IT gestützte – Dienstleistungen zu Produkten und Dienstleistungen des Unternehmens angesprochen werden. Wie in Kapitel 3.2.2.7 dargestellt ermöglicht eine Software auf Basis einer SOA auch, eine Unternehmens-IT portabel zu gestalten. Deshalb ist die Ansprache bestimmter Märkte – z. B. Vertretermärkte und mobile Kunden – auch möglich.

Eine Marktstrategie stellt die ‚Internationalisierung' dar.[824] Unternehmen verfolgen eine solche Strategie, um sich Zugang zu Ressourcen zu sichern, um Skaleneffekte und Verbundeffekte zu realisieren, um unterschiedliche Ausprägungen bei Faktorkosten ausnutzen und um erweiterte Möglichkeiten des Risikomanagements nutzen zu können. Internationalisierung stellt bestimmte Anforderungen an die Organisation,[825] z. B. an Organisationsstrukturen und Prozesse, und folglich auch an die Unternehmens-IT. Zu diesen Anforderungen gehört die Bildung von organisatorischen Aufbaustrukturen, die Internationalisierung unterstützen, die Koordination ausländischer Tochtergesellschaften und die Anpassung der Koordination an Auslandstätigkeiten.

Organisatorische Aufbaustrukturen, die Internationalisierung unterstützen, sind internationale Divisionen, die in folgende Varianten unterschieden werden können:[826] weltweite Gebietsdivisionen, Produktdivisionen und Matrixstrukturen. *Gebietsdivisionen* sind eigenständige Organisationen, die direkt der Konzernleitung unterstellt sind.[827] Die Unternehmens-IT kann in diesem Fall zwischen zwei Extremen ausgeprägt sein: zum einen

---

[822] Eine Produkt-Markt-Strategie kann auch auf Faktormärkte bezogen werden. In diesem Fall hat die Software z. B. auch Einfluss auf Faktormärkte und die Argumentation kann übertragen werden.

[823] Vgl. Dostal, Jeckle /Service-orientierte Architektur/ S. 53.

[824] Vgl. zum folgenden Absatz Kieser, Walgenbach /Organisation/ S. 284-285.

[825] Vgl. Kieser, Walgenbach /Organisation/ S. 289-309.

[826] Vgl. Kieser, Walgenbach /Organisation/ S. 289-293.

[827] Vgl. Kieser, Walgenbach /Organisation/ S. 292.

unterhält die Gebietsdivision eine eigenständige Unternehmens-IT sowie speziell gefertigte Programme, die primär der Berichterstattung an die Konzernleitung über die Unternehmens-IT der Konzernmutter dienen; zum anderen umfasst die Unternehmens-IT des Konzerns sowohl die Konzernmutter als auch die Gebietsdivisionen. Im ersten Fall ist eine direkte Unterstützung einer Unternehmens-IT durch den Einsatz einer SOA nicht erkennbar. Im zweiten Fall kann eine SOA die weltweit verteilte Unternehmens-IT unterstützen, da verteilte Systeme – somit auch eine verteilte Unternehmens-IT – gestärkt werden.[828] Genau dann ist eine konzernweit einheitliche Kommunikationsinfrastruktur vorhanden, die es allen Gebietsdivisionen ermöglicht, auf Dienstleistungen zurückzugreifen, die nur einmal angeboten und erstellt werden müssen. *Produktdivisionen* erhalten die weltweite Verantwortung für ein Produkt bzw. für eine Produktgruppe.[829] Hiermit kann eine Unternehmens-IT in ähnliche Extreme unterschieden werden. Zum einen existiert innerhalb einer Produktdivision eine eigenständige Unternehmens-IT abgegrenzt zu anderen Produktdivisionen des Unternehmens; zum anderen wird eine eigenständige Unternehmens-IT in entsprechenden Niederlassungen produkt-, aber nicht gebietsübergreifend aufgebaut. Diese Fälle entsprechen in der Argumentation der der Gebietsdivisionen. *Matrixstrukturen* stellen eine Kombination der Gebiets- und Produktdivisionsgliederung dar. Theoretisch kann auch eine Kombination der oben dargestellten Gliederungen einer Unternehmens-IT vorgenommen werden, und die getroffenen Argumentationen gelten entsprechend.

Die Koordination ausländischer Tochtergesellschaften geht einher mit einer hohen Zentralisierung der Koordination, die im internationalen Kontext auf Programmierung und Formalisierung hinausläuft.[830] Maßnahmen, die diesen Anforderungen gerecht werden[831] und durch eine Unternehmens-IT unterstützt werden können sind: Zentralisierung/Dezentralisierung von Entscheidungen und Programmen (Vorauskoordination).[832]

---

[828] Vgl. Kapitel 3.2.2.7.2.

[829] Vgl. Kieser, Walgenbach /Organisation/ S. 290-291.

[830] Vgl. Kieser, Walgenbach /Organisation/ S. 298-299.

[831] Vgl. Kieser, Walgenbach /Organisation/ S. 300.

[832] Kieser und Walgenbach identifizieren insgesamt neun Maßnahmen zur Koordination internationaler Aktivitäten. Maßnahmen, die nicht direkt durch eine Unternehmens-IT unterstützt werden sind beispielsweise ‚Persönliche Kontakte', ‚Sozialisation', ‚Planung und Kontrolle' und ‚Abteilungsbildung'. (Vgl. Kieser, Walgenbach /Organisation/ S. 300) Eine indirekte Unterstützung durch Bestandteile einer Unternehmens-IT wie z. B. ein unternehmensweites Adressbuch zur Pflege persönlicher Kontakte wird im Kontext dieser Arbeit ausgeklammert. Vgl. zur Gegenüberstellung

Eine SOA unterstützt diese Maßnahmen, da die Zentralisierung von Dienstleistungen in einer SOA durch das Anbieten eines Services für alle potentiellen Nutzer explizit vorgesehen ist. Gleichzeitig ist die entsprechend notwendige Dezentralisierung dadurch erreichbar, dass die Nutzung einer Dienstleistung weltweit und verteilt geschehen kann und Anpassungen einer Dienstleistung für lokale Zwecke unter Nutzung bestehender Services durchgeführt werden können. Vorauskoordination in Form von Programmen wird durch eine SOA unterstützt, weil durch das Anbieten von Services für das gesamte Unternehmen Vorgehensweisen und Arbeitsergebnisse zum Zwecke der Vorauskoordination erstellt werden können.

Des Weiteren stellen Auslandsaktivitäten unterschiedliche Anforderungen an die Koordination, die deswegen entsprechend angepasst werden muss.[833] Ergebnisse, die sich daraus optimalerweise ergeben (und die für eine Unternehmens-IT relevant sind), sind hybride Strukturen[834] oder integrierte Netzwerkstrukturen[835] sowie ein minimaler Einsatz technokratischer Koordinationsmechanismen.[836] Eine SOA unterstützt hybride Strukturen deswegen, weil Dienstleistungen in dem Unternehmen von den entsprechenden Stellen verwendet werden können, und zwar unabhängig davon, ob die Stellen weltweit verteilt arbeiten und ob diese in unterschiedliche Unternehmensteile strukturiert sind. Integrierte Netzwerke werden deswegen unterstützt, weil Unternehmensteile unabhängig voneinander agieren können und gleichzeitig die Möglichkeit besitzen, die gemeinsam nutzbaren Teile einer Unternehmens-IT einheitlich und redundanzfrei nutzen zu können.

Insgesamt wird die Unterstützung von Gesamtunternehmensstrategien positiv bewertet.

---

der Vorauskoordination zum Gegensatz ‚Feedbackkoordination' und zur Darstellung potentieller Vorteile Mellis /Projektmanagement/ S. 136-137 oder Kieser, Walgenbach /Organisation/ S. 105-109.

833 Vgl. Kieser, Walgenbach /Organisation/ S. 301-309.

834 Unter hybriden Strukturen verstehen Kieser und Walgenbach gemischte organisatorische Strukturen. Beispielsweise das Existieren sowohl internationaler Gebiets- als auch Produktdivisionen, die jeweils auch Teilaufgaben für andere Divisionen übernehmen können. Vgl. Kieser, Walgenbach /Organisation/ S. 304-305.

835 Integrierte Netzwerkstrukturen stellen eine Struktur dar, in der einzelne weltweit verteilte Unternehmensteile bestimmte Aufgaben dem örtlichen Umfeld und den entsprechenden Kompetenzen angepasst erhalten. Dadurch entstehen sog. ‚Corporate Centers of Excellence', die für bestimmte Aufgaben eine u. U. alleinige Kompetenz im Unternehmen erhalten. Vgl. Bartlett /Building/ S. 382 zitiert nach Kieser, Walgenbach /Organisation/ S. 306-307.

836 Vgl. Kieser, Walgenbach /Organisation/ S. 304-307.

Unterstützung von Wettbewerbsstrategien

Unterstützung von Wettbewerbsstrategien ist die Fähigkeit einer Softwarearchitektur, eine Strategie der Kostenführerschaft, der (Produkt-)Differenzierung und eine Nischenstrategie unterstützen zu können.

Verschiedene Wettbewerbsstrategien sind nach Berensmann in den drei Bereichen Ressourcenzugriff, Zugang zu Absatzmärkten und Geschäftsprozesse möglich.[837] Die Strategiegestaltung in den Bereichen der Faktoren- und Absatzmärkte werden in dieser Arbeit (Macharzina folgend) im Bereich der Produkt-Markt-Strategien untersucht und wurden somit schon bewertet.[838] Heute verlieren die Differenzierungsmöglichkeiten im Ressourcenzugriff und dem Zugang zu Absatzmärkten in diesem Bezug ihre strategische Bedeutung.[839] Der Wandel von einer historisch datenorientierten hin zu einer prozessorientierten Betrachtung muss heute dann erfolgen, wenn Unternehmen die Effektivität durch eine Fokussierung auf die technische Unterstützung betrieblicher Prozesse steigern wollen.[840] Auch deswegen steigt die strategische Bedeutung der Geschäftsprozesse um ein Vielfaches.[841]

Nach Porter werden die Kosten, die im Rahmen der Wettbewerbsstrategie **Kostenführerschaft**[842] beeinflusst werden können, von zehn Kostentreibern festgelegt: Skaleneffekte (Economy of scales), Lerneffekte (Learning), Grad der Kapazitätennutzung (Pattern of capacity utilization), Verflechtungen (Linkages), interne wechselseitige Beziehungen (Interrelationships), Integration (Integration), zeitliche Abstimmung (Timing), Handlungsfreiheit (Discretionary policies), Ort (Location) und institutionelle Faktoren (Institutional factors).[843] *Lerneffekte*[844] können in kurzer Zeit ausgenutzt werden, weil eine SOA eine hohe Wandlungsfähigkeit aufweist[845] und dadurch ein schnel-

---

[837] Vgl. Berensmann /IT matters/ S. 274-276.

[838] Siehe weiter oben im gleichen Kapitel.

[839] Vgl. Berensmann /IT matters/ S. 274-276.

[840] Vgl. Lublinsky, Tyomkin /Dissecting SOA/ S. 54.

[841] Vgl. Berensmann /IT matters/ S. 274-276.

[842] Vgl. Porter /Advantage/ S. 62-118.

[843] Vgl. Porter /Advantage/ S. 70-84.

[844] Vgl. Porter /Advantage/ S. 73-74.

[845] Vgl. Kapitel 3.2.2.2.

ler Einsatz des Erlernten erreicht werden kann. Der *Grad der Kapazitätsnutzung*[846] kann durch eine SOA erhöht werden, weil die Kosten für implementierte Services unternehmensweit und auch über Unternehmensgrenzen hinweg auf mehrere Dienstleistungsnachfrager verteilt werden können. Die Kosten klassischer Anwendungen einer Unternehmens-IT dagegen können i. d. R. nur auf einen eindeutig benannten, ggf. unveränderlichen Nutzerkreis wie z. B. eine Abteilung aufgeteilt werden. *Verflechtungen*[847] werden bei Porter in Verflechtungen innerhalb der Wertschöpfungskette und auf vertikaler Ebene unterschieden. Den Kostentreiber ‚Verflechtung innerhalb einer Wertschöpfungskette' unterstützt eine SOA nicht eindeutig positiv oder negativ. ‚Vertikale Verflechtung' jedoch unterstützt eine SOA insofern sehr gut, weil sie die Möglichkeit bietet, externe Systeme auch nachträglich anzubinden.[848] *Integration*[849] wird dadurch unterstützt, weil die Wahlfreiheit besteht, ob Integration stattfinden soll. Erstens weil eine SOA die Fähigkeit der Erweiterung für eine im Zusammenhang mit Insourcing relevante Erweiterung der Unternehmens-IT unterstützt; zweitens, weil eine Anbindung externer Systeme möglich ist.[850] Der Kostentreiber *Timing*[851] wird durch eine positive Wandlungsfähigkeit[852] einer SOA positiv beeinflusst, weil schnelles Agieren und damit potentiell auch die Realisierung von „First-Mover-Effekten"[853] möglich ist. *Handlungsfreiheiten*[854] werden in bestimmten Bereichen indirekt durch eine SOA unterstützt. Sobald Handlungsfreiheiten bestehen, die auch auf die Unternehmens-IT Einfluss haben, hierunter fallen z. B. Lieferzeiten, Vertriebskanäle und Produkte (z. B. auch Produktveredelungen[855]), unterstützt die positiv bewertete Wandlungsfähigkeit[856] die Fähigkeit

---

[846] Vgl. Porter /Advantage/ S. 74-75.

[847] Vgl. Porter /Advantage/ S. 75-78.

[848] Vgl. Kapitel 3.2.2.2.1 und 3.2.2.1.1.

[849] Unter Integration in Bezug auf die Wettbewerbsstrategie Kostenführerschaft versteht Porter die Entscheidung für eine bestimmte Ausprägung des In- oder Outsourcings. Vgl. Porter /Advantage/ S. 79.

[850] Vgl. Kapitel 3.2.2.2.1 und 3.2.2.1.1.

[851] Vgl. Porter /Advantage/ S. 79-80.

[852] Vgl. Kapitel 3.2.2.2.

[853] Vgl. Porter /Advantage/ S. 79-80.

[854] Unter Handlungsfreiheiten versteht Porter die Freiheiten, die bleiben, wenn Richtlinien eines Unternehmens beachtet werden. Unter hierfür relevante Richtlinien fallen beispielsweise der Service, welcher geboten werden soll, oder Mitarbeitervorteile, die gewährt werden, um am entsprechenden Markt bestehen und attraktiv sein zu können. Vgl. Porter /Advantage/ S. 80-82.

[855] Vgl. Porter /Advantage/ S. 131-132 & S. 137-138.

[856] Vgl. Kapitel 3.2.2.2.

der Unternehmens-IT, die Handlungsfreiheiten zu nutzen. Handlungsfreiheiten werden durch eine SOA jedoch auch eingeschränkt: durch die Wahl einer SOA findet eine Festlegung auf eine Kommunikationsinfrastruktur und damit auf die dabei verwendeten Kommunikationsmethoden statt. Eine SOA schränkt z. B. die Freiheit ein, beliebig umfangreiche Datenmengen zwischen Services zu transferieren und gleichzeitig die bisher erreichte Datenverarbeitungsmenge zu halten, weil die Kommunikation i. d. R. einen hohen Overhead (vgl. Netto- vs. Bruttovolumen[857]) beinhaltet. Der Kostentreiber *Ort*[858] wird durch eine SOA unterstützt, weil eine verteilte Unternehmens-IT unterstützt wird[859] und nicht für jeden Standort eigene Software betrieben und erstellt werden muss, sondern Software anderer Standorte (bzw. über ein gemeinsames IT-Infrastrukturzentrum) genutzt werden kann. Demzufolge können Ortsentscheidungen, die zu Wettbewerbsvorteilen führen sollen, unabhängig von Gesichtspunkten der Unternehmens-IT getroffen werden. Institutionelle Faktoren[860] liegen i. d. R. außerhalb der Kontrolle eines Unternehmens. Deswegen können Unternehmen lediglich versuchen, diese zu beeinflussen oder das Unternehmen so zu gestalten, dass die entsprechenden Auswirkungen minimiert werden. Eine SOA unterstützt die Anstrengungen, entsprechende Auswirkungen zu minimieren, weil eine SOA bei der Wandlungsfähigkeit[861] und Interoperabilität[862] positiv bewertet wurde. Folglich können Änderungen und neue Anforderungen, die z. B. aus der Gesetzeslage oder steuerlichen Bedingungen resultieren, schnell umgesetzt werden. Zu den anderen verbleibenden Kostentreibern wie Skaleneffekte, Lerneffekte und interne wechselseitige Beziehungen, konnte kein Einfluss einer Softwarearchitektur aufgestellt werden.

Zusammenfassend unterstützt eine SOA die Wettbewerbsstrategie Kostenführerschaft und somit auch die Aktivitäten,[863] die im Rahmen dieser Strategieverfolgung notwendig werden können.

---

[857] Vgl. Kapitel 3.1.3.4.1.

[858] Vgl. Porter /Advantage/ S. 82-83.

[859] Vgl. Kapitel 3.2.2.7.2.

[860] Unter ‚Institutionellen Faktoren' versteht Porter Faktoren wie beispielsweise die Gesetzeslage, Steuerbedingungen und Gewerkschaftsstruktur. Vgl. Porter /Advantage/ S. 83-84.

[861] Vgl. Kapitel 3.2.2.2.

[862] Vgl. Kapitel 3.2.2.1.1.

[863] Porter /Advantage/ S. 99- 106.

Die zweite Wettbewerbsstrategie, die untersucht wird, ist die **(Produkt-) Differenzierung**.[864] Differenzierung kann überall in der Wertschöpfungskette und nicht nur in Bezug auf das Produkt und auf Marketingmaßnahmen eingeführt werden.[865] Hierunter fallen z. B. die Auswahl von Rohstoffen, Gestaltung des Supports, die potentielle Reichweite ihrer Aktivitäten und die Gestaltung der Vertriebskanäle.[866] Weil Differenzierung über die gesamte Wertschöpfungskette erreicht werden kann, stimmen die potentiellen Gestaltungsbereiche zur Differenzierung mit denen der Kostenführerschaft zum großen Teil überein.[867] Die Gestaltungsbereiche der Differenzierung betreffen Skaleneffekte (Economy of scales), Lerneffekte (Learning), Verflechtungen (Linkages), interne wechselseitige Beziehung (Interrelationships), Handlungsfreiheit (Discretionary policies), Integration (Integration), zeitliche Abstimmung (Timing), Ort (Location) und institutionelle Faktoren (Institutional factors).[868] Wie oben schon gezeigt wurde, unterstützt eine SOA die Gestaltung in diesen Bereichen.

Ein weiterer Punkt der Wettbewerbsstrategie Differenzierung ist, dass die getroffenen Gestaltungsentscheidungen für das gesamte Unternehmen konsistent und koordiniert ausgeführt werden.[869] Eine Unternehmens-IT auf Basis einer SOA unterstützt diesen Punkt deswegen, weil die Bestandteile der Differenzierungsstrategie, die die Unterstützung durch die Unternehmens-IT benötigen, unternehmensweit einheitlich angeboten werden können.[870] Die Koordination der Aktivitäten wird unterstützt, weil Änderungen an zentraler Stelle – den Services – für das gesamte Unternehmen durchgeführt werden können. Zusammenfassend unterstützt eine SOA die Wettbewerbsstrategie Differenzierung und die Aktivitäten, die aus den Gestaltungsentscheidungen resultieren.

Die dritte Wettbewerbsstrategie, die zu diesem Unterqualitätsattribut untersucht werden soll, ist die **Nischenstrategie**[871]. Hierbei fokussiert ein Unternehmen seine Aktivitäten

---

[864] Vgl. Porter /Advantage/ S. 119-163.

[865] Vgl. Porter /Advantage/ S. 119.

[866] Vgl. Porter /Advantage/ S. 120-123.

[867] Nur der Grad der Kapazitätennutzung (pattern of capacity utilization) wird von Porter nicht als relevant für eine Differenzierungsstrategie angesehen.

[868] Vgl. Porter /Advantage/ S. 124-127.

[869] Vgl. Porter /Advantage/ S. 123.

[870] Vgl. in diesem Kapitel (3.2.3.1.1) weiter oben.

[871] Vgl. Porter /Advantage/ S. 231-272.

auf ein eindeutig abgegrenztes Industriesegment.[872] Die Segmentierung wird dabei auf Basis von Produktunterschieden, Käuferprofilen, Vertriebskanälen und geographischer Lage der Käufer vorgenommen.[873] Eine Nischenstrategie steht für die Anwendung der Wettbewerbsstrategien Kostenführerschaft, Differenzierung oder sogar beider strategischer Vorgehensweisen – fokussiert auf ein eindeutig abgegrenztes Marktsegment. Das Ziel ist, dieses Marktsegment in der entsprechenden Strategie noch effektiver und/oder effizienter bedienen zu können als ein Wettbewerber, der den gesamten Markt anspricht.[874] Durch diese Fokussierung kann für die meisten Industriesegmente abgeleitet werden, dass die oben genannte Ausprägung der Fähigkeit zur Unterstützung einer Kostenführerschaft und Differenzierung ebenfalls auf die Nischenstrategie übertragbar ist. Die einzigen Kriterien einer potentiellen Segmentierung, auf die eine Softwarearchitektur Einfluss haben kann, sind Funktionen (Features), Käuferstrategien (Buyer's strategy) und vertikale Integration (Vertical integration).[875] Einfluss auf *Funktionen* besteht deshalb, weil die Funktionalität eines Produktes über Software erweitert werden kann. Beispielsweise könnte eine Dienstleistung, die ein Produkt erweitert, ebenso eine Dienstleistungen auf Softwarebasis darstellen. Bekannte Produkterweiterungen sind z. B. Staudaten in Echtzeit für Navigationssysteme oder die Funktion von digitalen Videorekordern, um Werbung beim Aufnehmen von Fernsehaufnahmen nicht mit aufzunehmen. In diesem Fall bietet eine SOA Unterstützung, weil die Produkte i. d. R. an unterschiedlichen Orten durch die Kunden genutzt werden und solche verteilte Strukturen unterstützt werden können.[876] *Käuferstrategien* sind die entsprechenden Wettbewerbsstrategien gegenüber Käufergruppen.[877] Sie werden in diesem Zusammenhang aus Sicht der Argumentation zu den Wettbewerbsstrategien Kostenführerschaft und Differenzierung (siehe oben) betrachtet. Hier gelten die gleichen Argumentationen wie oben, lediglich in umgekehrter Sichtweise. Beispielsweise kann eine *(vertikale) Integration* auch dann

---

[872] Vgl. Porter /Advantage/ S. 232.

[873] Vgl. Porter /Advantage/ S. 237-238.

[874] Vgl. Porter /Strategy/ S. 38-40.

[875] Vgl. zu weiteren Kriterien einer Segmentierung Porter /Advantage/ S. 237-248. Porter unterteilt die Kriterien in die vier oben vorgestellten Bereiche (Produktunterschiede, Käuferprofile, Vertriebskanäle und geographische Lage). Kriterien, zu denen beispielsweise kein Einfluss einer SOA gefunden werden konnte sind Produktgröße, Preisgestaltung, Finanzkraft, Besitzerstruktur der Unternehmen, geographisches Klima und Grund des Produktkaufes.

[876] Vgl. in diesem Kapitel (3.2.3.1.1) weiter oben.

vorgenommen werden, wenn die Unternehmens-IT des Lieferanten die Integration in die Unternehmens-IT des Käufers unterstützt.

Insgesamt unterstützt eine Unternehmens-IT auf Basis einer SOA umfassend Wettbewerbsstrategien. Die Unterstützung von Wettbewerbsstrategien wird positiv bewertet.

Unterstützung des Strategiewandels

Unterstützung des Strategiewandels ist die Fähigkeit einer Softwarearchitektur, den Strategiewandel eines Unternehmens zu unterstützen.

Strategiewandel kann sich auf den Wandel von allen im Unternehmen identifizierbaren Strategietypen beziehen. Softwarearchitekturen haben Einfluss auf die Unterstützung des Strategiewandels, wenn der Wandel Auswirkungen auf die Unternehmens-IT hat. Dies kann z. B. dann gegeben sein, wenn die neue Strategie neue strukturelle und funktionale Anforderungen an die Unternehmens-IT stellt. Strukturelle Anforderungen sind z. B. dann gegeben, wenn ein Unternehmen die strategische Ausrichtung ihres ‚Ortes' ändert, wohingegen neue funktionale Anforderungen dann gegeben sein können, wenn Produkte (z. B. über zusätzliche Supportleistungen) durch die Unternehmens-IT aufgewertet werden sollen. Wie schon gezeigt werden konnte, unterstützt eine SOA sowohl strukturelle als auch funktionale Änderungen einer Unternehmens-IT in hohem Maß.[878] Weil keine weiteren Einflüsse des Strategiewandels auf eine Unternehmens-IT identifiziert werden können,[879] wird die Unterstützung des Strategiewandels positiv bewertet.

3.2.3.1.2 Nutzen der Strategieunterstützung

Einschränkungen einer Software bestehen für einen Unternehmensgestalter, wenn die Software entsprechende Strategien und einen Wandel zu entsprechenden Strategien nicht unterstützt. Dieses Bewertungskriterium wird im nächsten Kapitel bei der Bewertung der Unterqualitätsattribute eingehend untersucht.

---

[877] Käuferstrategien beziehen sich hier nur auf die Käufergruppen der industriellen und kommerziellen Käufer. Vgl. Porter /Advantage/ S. 241-242.

[878] Vgl. Kapitel 3.2.2.2.

[879] Weitere Aktivitäten und Auswirkungen des *Strategiewandels* sind z. B. die Formulierung von Strategien und die Analyse von Strategien. Diese beeinflussen eine Unternehmens-IT nicht. Die Umsetzung von Strategien dagegen besitzt die im Text dargestellten Einflüsse auf die Unternehmens-IT.

Über diese Bewertung hinaus kann beurteilt werden, inwiefern eine SOA den Prozess der Strategieumsetzung unterstützt. Eine Festlegung von Geschäftsprozessen erfolgt stets unter Berücksichtigung strategischer Ziele.[880] Damit ein Unternehmen seine strategischen Ziele verfolgen kann, müssen entsprechend erstellte Geschäftsprozesse wie vorgesehen ablaufen können. Weil ein Unternehmen heute zur Unterstützung seiner Geschäftsprozesse auf Informationssysteme angewiesen ist,[881] ergibt sich aus strategischer Sicht die Anforderung an die Unternehmens-IT, die Geschäftsprozesse eines Unternehmens exakt abbilden zu können, da die Verfolgung der strategischen Ziele andernfalls durch ein Abweichen von den vorgesehenen organisatorischen Abläufen gefährdet ist. Wie in Kapitel 3.1.3 argumentiert, ist die Bewertung der Benutzbarkeit (aus Entwicklersicht) und die Funktionserfüllung positiv. Die Schlussfolgerung ist, dass eine Software auf Basis einer SOA den Prozess der Strategieumsetzung dadurch unterstützt, weil die Geschäftsprozesse eines Unternehmens abgebildet werden und somit auch die strategischen Ziele des Unternehmens verfolgt werden können.

Wie in Kapitel 3.1.4.2.3 aufgezeigt, ist heute die Strategieunterstützung vor allem auch deswegen relevant, da Unternehmensaufkäufe oder -zusammenlegungen (im Folgenden zusammenfassend Fusionen genannt) in der Praxis häufig vorkommen und ein fester Bestandteil strategischer Planung geworden sind.[882] Fusionen bedingen eine Integration der Unternehmens-IT-Systeme aller beteiligten Unternehmen.[883] Schlüsselaktivitäten, die die Unternehmens-IT betreffen und die für einen erfolgreichen Firmenzusammenschluss notwendig sind, umfassen das Aufstellen einer zusammenhängenden Integrationsstrategie, Kommunikation und Schnelligkeit der Umsetzung.[884] Bei allen drei Schlüsselaktivitäten müssen technologische Aspekte berücksichtigt werden: die Integrationsstrategie muss Möglichkeiten des technologischen Zusammenwirkens beachten, Kommunikation findet heute vielfältig über Unternehmens-IT statt, und die Unterneh-

---

[880] Vgl. Kirchmer /Einführung/ S. 38.

[881] Vgl. Kapitel 1.1.2.

[882] Vgl. Trapp, Otto /Einsatzmöglichkeiten/ S. 102.

[883] Vgl. Trapp, Otto /Einsatzmöglichkeiten/ S. 105-108.

[884] Insgesamt identifiziert Epstein fünf Schlüsselfaktoren, die Weiteren sind: ein starkes Integrationsteam und angepasste Maßstäbe. Vgl. dazu und zum folgenden Absatz Epstein /Drivers/ S. 176-179; Trapp und Otto identifizieren nicht explizit Schlüsselfaktoren, sondern argumentieren, dass die IT-Strategie (S. 104) einen Firmenzusammenschluss unterstützen muss, dass Kommunikation mittels durchgängiger Informationsflüsse (S. 103) erreicht werden muss, und sie betonen im gesamten Arti-

mens-IT muss die jeweilig gewünschte Integrationsgeschwindigkeit unterstützen können. Unter Beachtung dieser Anforderungen bei Merger & Acquisitions-Strategien unterstützt eine Softwarearchitektur nach serviceorientierten Gestaltungsprinzipien einen Firmenzusammenschluss.[885] Argumentieren lässt sich dies auch, weil der Unternehmensgestalter die Formulierung einer Integrationsstrategie losgelöst von der Ausprägung der Unternehmens-IT treffen kann. Sowohl verschiedene Strategien als auch ein potentieller Strategiewandel werden dabei unterstützt.[886] Kommunikation zwischen den Unternehmen wird durch die positive Bewertung der Interoperabilität[887] unterstützt, und durch die positiv bewertete Wandlungsfähigkeit[888] und Benutzbarkeit aus Entwicklersicht[889] ist die Unterstützung auch hoher Integrationsgeschwindigkeiten möglich.

3.2.3.1.3 Gesamtbewertung der Strategieunterstützung

Zusammenfassend wird das Qualitätsattribut ‚Strategieunterstützung' positiv bewertet (Tab. 3-18).

(Unter-)Qualitätsattribute	***Wirkung***
Prozess der Strategieumsetzung	+
Merger & Acquisitions-Strategien	+
Unterstützung von Gesamtunternehmensstrategien	+
Unterstützung von Wettbewerbsstrategien	+
Unterstützung des Strategiewandels	+
Gesamtbewertung	+

Tab. 3-18: Nutzenbewertung des Qualitätsattributes Strategieunterstützung

kel den (Kosten sparenden) kurzen Zeitfaktor, vgl. Trapp, Otto /Einsatzmöglichkeiten/; Vgl. des Weiterhin Keller /EAI/ S. 13-14; Kaib /EAI/ S. 43.

[885] Vgl. Aier, Schönherr /Flexibilisierung/ S. 13.

[886] Vgl. Kapitel 3.2.3.1.2

[887] Vgl. Kapitel 3.2.2.1.1.

[888] Vgl. Kapitel 3.2.2.2.

[889] Vgl. Kapitel 3.2.2.3.

3.2.3.2 Nachhaltigkeit

Nachhaltigkeit ist die Fähigkeit einer Unternehmens-IT, in mindestens ausreichendem Maße dauerhaft zukunftsfähig zu sein. Nachhaltigkeit ist dann gegeben, wenn eine Unternehmens-IT strukturanaloge und autopoietische Merkmale aufweist.[890]

Im Folgenden werden auf Basis von Argumentationen Bewertungen der Unterqualitätsattribute der Nachhaltigkeit vorgenommen. Anschließend werden die Wirkungen einer SOA auf die Nachhaltigkeit einer Unternehmens-IT untersucht und es wird eine Gesamtbewertung aufgestellt.

3.2.3.2.1 Nutzen der Unterqualitätsattribute der Nachhaltigkeit

Die Unterqualitätsattribute der Nachhaltigkeit sind Strukturanalogie und Autopoiesis.[891]

Strukturanalogie

Wie in Kapitel 3.1.4.3.1 vorgestellt identifiziert Gronau vier grundlegende Reorganisationsansätze[892] und stellt drei Prinzipien auf, welche die Reorganisationsansätze unterstützen. Dies sind die Prinzipien der Dezentralität, der Flexibilität und Dynamik sowie der strukturellen Analogie.[893] Das Unterqualitätsattribut Strukturanalogie wird anhand dieser Prinzipien bewertet.

Das *Prinzip der Dezentralität* betrifft Strukturen und Prozesse sowohl auf zeitlicher als auch auf räumlicher Ebene. Eine SOA ermöglicht verteilte Systeme.[894] Dadurch kann eine Unternehmens-IT alle Anforderungen der Dezentralität von Strukturen und Prozessen auf räumlicher Ebene unterstützen. Eine räumliche Festlegung ist nicht notwendig, weil die Kommunikationsinfrastruktur zum Ausführungszeitpunkt einer Dienstleistung ad hoc bestimmen kann, an welcher Stelle die Dienstleistung erbracht werden soll bzw.

---

[890] Vgl. Kapitel 3.1.4.3.2.

[891] Vgl. Kapitel 3.1.4.3.2.

[892] Dies sind die Reorganisationsansätze ‚Segmentierung', ‚Prozessorientierung', ‚kontinuierliche Reorganisation' sowie ‚Auflösung von Unternehmensgrenzen'. Vgl. Gronau /Wandlungsfähige Informationssystemarchitekturen/ S. S. 60-64.

[893] Vgl. zum folgenden Absatz Gronau /Wandlungsfähige Informationssystemarchitekturen/ S. 216-217.

[894] Vgl. Kapitel 3.2.2.7.2.

kann. Durch die Fähigkeit einer SOA, asynchrone Prozesse zu unterstützen,[895] können Anforderungen der Dezentralität von Prozessen auch auf zeitlicher Ebene erfüllt werden, weil Prozesse eines Geschäftsprozesses nicht in chronologisch festgelegter Reihenfolge bearbeitet werden müssen, sondern zu unterschiedlichen Zeitpunkten in jeweils wechselnder chronologischer Reihenfolge bearbeitet werden können.

Das *Prinzip der Flexibilität und Dynamik* betrifft die Schaffung und Aufrechterhaltung der Unterstützung von Geschäftsprozessen sowie die Unterstützung von Veränderungen der unternehmensinternen und -übergreifenden Strukturen (wie beispielsweise Allianzen) durch die Unternehmens-IT. Wie von Carlson und Tyomkin beschrieben wurde, kann eine SOA – durch die Erhöhung der Wandlungsfähigkeit[896] einer Unternehmens-IT – den Anforderungen einer sich schnell wandelnden betrieblichen Umgebung Rechnung tragen.[897] Dies kann dadurch erreicht werden, weil neue Geschäftsprozesse, Änderungen von Geschäftsprozessen und Erweiterungen der Unternehmens-IT durch Änderungen der zu unterstützenden Unternehmensstrukturen gut umgesetzt werden können. Es kann davon ausgegangen werden, dass die positiv bewertete Wandlungsfähigkeit eine schnelle und umfassende Wandlung der Unternehmens-IT unterstützt.

Das *Prinzip der strukturellen Analogie* betrifft das Merkmal, dass sich eine Unternehmens-IT harmonisch an den jeweils vorherrschenden Organisationstyp anpassen kann.[898] Gronau erbrachte in seiner Arbeit den indizienbasierten Nachweis[899] der Existenz solcher strukturanalogen Paare[900] und identifiziert z. B. die Paare ‚segmentierte Organisationsform' und ‚objektorientiertes Informationssystem' sowie ‚prozessorientierte Organisationsform' und ‚Workflow-Managementsysteme'.[901] Das Prinzip der

---

[895] Asynchroner Ablauf von Services kann eine Ausprägung einer SOA sein, vgl. Kapitel 2.2.4.3.

[896] Vgl. Kapitel 3.2.2.2.

[897] Vgl. Carlson, Tyomkin /Good Design/ S. 13.

[898] Den Nachweis, dass eine solche Anpassung zwischen Organisationsentwicklung und Unternehmens-IT stattfindet, erbringt Gronau auch in seiner Arbeit. Vgl. Gronau /Wandlungsfähige Informationssystemarchitekturen/ S. 217-218.

[899] Gronau stellt in seiner Arbeit weder eine empirische Studie vor noch argumentiert er auf umfassender theoretischer Basis, sondern stellt indizienhaft alltäglich beobachtbare Beispiele zusammen und unterstreicht diese mit der Arbeit von Krahe, der Ähnlichkeiten im Prozessmanagement als organisatorischem Ansatz und Informationstechnologie untersucht hat. Vgl. Gronau /Wandlungsfähige Informationssystemarchitekturen/ S. 218 in Verbindung mit Krahe /Prozessmanagement/ S. 4.

[900] Vgl. Gronau /Wandlungsfähige Informationssystemarchitekturen/ S. 217-218.

[901] Vgl. Gronau /Wandlungsfähige Informationssystemarchitekturen/ S. 217.

strukturellen Analogie kann auf Basis der Organisationstheorie, d. h. aufbauend auf der Aufbau- und Ablauforganisation eines Unternehmens untersucht werden.[902]

Organisatorische Strukturen der Aufbauorganisation sind durch viele Autoren untersucht worden. Ein umfassendes und anerkanntes Verständnis der Strukturierung von Unternehmen hat sich gebildet.[903] Mellis nimmt eine Unterteilung der Strukturierungsdimensionen in horizontale bzw. vertikale Arbeitsteilung und Koordination vor, die für diese Arbeit als Bewertungskriterien geeignet sind.[904] Im Zuge einer horizontalen Arbeitsteilung erfolgt die Zerlegung einer Gesamtaufgabe in Teilaufgaben, die durch unterschiedliche organisatorische Stellen bearbeitet werden (sog. Spezialisierung).[905] Eine SOA unterstützt eine *horizontale Arbeitsteilung* dadurch, weil erstens die Aufteilung einer Gesamtaufgabe in Teilsaufgaben durch die Unternehmens-IT in Form von Services für jede Teilaufgabe abgebildet werden kann;[906] zweitens ist die Unterstützung durch die Unternehmens-IT auch unabhängig vom Standort des Nachfragers einer Dienstleistung anhand der Verwendung der Kommunikationsinfrastruktur[907] möglich.[908] Die Bewertung der Unterstützung einer *vertikalen Arbeitsteilung* – insbesondere Konfiguration (das Leitungssystem) und Entscheidungsdelegation – ist zur Bewertung einer SOA nicht relevant. Die vertikale Arbeitsteilung determiniert die Struktur der Wei-

---

902 Gronau nennt explizit diese beiden Kriterien als Beleg für das Prinzip struktureller Analogie. Geschäftsprozess bzw. Ablauforganisation und bezieht sich dabei auf Krahe, der nennt dies Prozessmanagement

903 Vgl. beispielsweise Kieser, Walgenbach /Organisation/; Frese /Organisation/ und Mintzberg /Structuring/.

904 Mellis identifiziert in seiner Arbeit Gestaltungsbereiche für die Softwareentwicklung und identifiziert dabei auch organisatorische Gestaltungsbereiche (in der Abgrenzung zu fachlich-technischen Gestaltungsbereichen). Er betrachtet in seiner Arbeit einen Typus eines Unternehmens: einen Softwareproduzenten. Diese von Mellis verwendeten Dimensionen einer Organisationsstruktur können stellvertretend für die Betrachtung in dieser Arbeit herangezogen werden, da sich die organisatorischen Dimensionen nicht von denen anderer Autoren unterscheiden. Vielmehr bringt die Strukturierung der Dimensionen nach Mellis für diese Arbeit Vorteile, indem eine Aggregation der in der Literatur diskutierten Dimensionen (vgl. beispielsweise Kieser, Walgenbach /Organisation/ S. 77-78 mit anschließend detaillierter Erläuterung auf den Seiten 78-176) vorgenommen wurde und dadurch in dieser Arbeit eine stärkere Fokussierung auf Kernaspekte vorgenommen werden kann. Somit muss nicht auf jede einzelne Dimension anderer Autoren eingegangen werden. Auch müssen die Dimensionen anderer Modelle namhafter Autoren nicht untersucht und ggf. aggregiert werden, weil sich eine identische Aggregation ergeben würde. Z. B. fasst Mellis die zwei Dimensionen ‚Konfiguration' und ‚Entscheidungsdelegation' (in diesem Fall nach Kieser, Walgenbach /Organisation/ S. 77) zu der Dimension ‚Vertikale Arbeitsteilung' zusammen (vgl. Mellis /Projektmanagement/ S. 161).

905 Vgl. Mellis /Projektmanagement/ S. 111-114.

906 Vgl. Kapitel 2.2.4.1.

907 Vgl. Kapitel 2.2.4.3.

908 Vgl. Kapitel 3.2.3.1.1.

sungsbefugnisse und der Aufgabeninhalte eines arbeitsteiligen Unternehmens. Die Ausprägungen der Weisungsbefugnisse und der Aufgabeninhalte eines Unternehmens werden durch eine Softwarearchitektur nicht beeinflusst, eine Unternehmens-IT muss diese vielmehr funktional abbilden können. Somit stellt die Art der vertikalen Arbeitsteilung eine funktionale Anforderung an eine Unternehmens-IT dar und kann nicht direkt durch eine Softwarearchitektur beeinflusst werden.[909] Nachteilig wirkt sich eine SOA jedoch auch nicht auf Gestaltungsalternativen einer vertikalen Arbeitsteilung aus. Es kann davon ausgegangen werden kann, dass funktionale Anforderungen der vertikalen Arbeitsteilung nicht durch die Art der Softwarearchitektur beeinflusst werden, weil diese einen Typus funktionaler Anforderungen darstellen. Koordination kann durch vier Koordinationsmechanismen umgesetzt werden:[910] Koordination (1) durch persönliche Weisung, (2) durch Selbstabstimmung, (3) durch Pläne und (4) durch Programme. Eine Softwarearchitektur hat keine Auswirkung auf die beiden ersten Koordinationsmechanismen (persönliche Weisung und Selbstabstimmung), weil eine entsprechende Unterstützung dieser Koordinationsmechanismen synonym der vertikalen Arbeitsteilung (s. o.) als funktionale Anforderungen an eine Unternehmens-IT formuliert werden müssen. Allerdings kann eine SOA die Koordination durch Pläne und Programme unterstützen. Koordination durch Pläne und Programme wird erreicht, indem festgelegt wird, wie bestimmte Aktivitäten durchgeführt werden sollen, z. B. durch die Vorgabe von Standardverfahren.[911] Solche Vorgaben können deswegen unterstützt werden, weil eine SOA die einheitliche Festlegung solcher Pläne und Programme in Services wartbar bietet, und zwar erstens durch eine Unterstützung verteilter Systeme[912] unternehmensweit einheitlich und zweitens durch die Verwendung einer Kommunikationsinfrastruktur[913] an zentraler Stelle.

---

[909] Vgl. Kapitel 3.1.3.2.1.

[910] Vgl. Mintzberg /Structuring/ S. 3-5; Kieser, Walgenbach /Organisation/ S. 108; Mellis /Projektmanagement/ S. 142-159.

[911] Vgl. Mellis /Projektmanagement/ S. 147-159. Koordination durch Pläne unterscheidet sich von der durch Programme durch einen unterschiedlichen zeitlichen Planungshorizont: Programme haben langfristigen Charakter, wohingegen von Plänen gesprochen wird, wenn zeitlich befristete Vorgaben existieren, die weder durch persönliche Weisung noch durch Selbstabstimmung erreicht wurden. Vgl. Mellis /Projektmanagement/ S. 156.

[912] Vgl. Kapitel 3.2.2.7.2.

[913] Vgl. Kapitel 2.2.4.3.

Nach Gaitanides beschreibt die *Ablauforganisation* eines Unternehmens die zeitliche und räumliche Gestaltung der Teilaufgaben und ihre Zusammenfassung zu Prozessen.[914] Darauf basierend beschreibt Mellis ein Konzept der Ablauforganisation, wonach die Ablauforganisation prozess- bzw. objektorientiert oder nach Rang orientiert gestaltet werden kann.[915] Eine prozessorientierte Ablauforganisation strukturiert die Prozesse eines Unternehmens nach den Tätigkeiten, die zur Leistungserbringung durchgeführt werden müssen. Diese Tätigkeiten werden von einer organisatorischen Einheit für alle Produkte bzw. Dienstleistungen innerhalb des Unternehmens ausgeführt. Eine objektorientierte Ablauforganisation dagegen strukturiert die Prozesse nach den Produkten bzw. Dienstleistungen eines Unternehmens. Hierbei ist eine organisatorische Stelle für alle Tätigkeiten verantwortlich, die zur Leistungserbringung für das jeweilige Produkt bzw. die jeweilige Dienstleistung durchgeführt werden müssen. Eine Strukturierung der Ablauforganisation nach Rang stellt die Delegationsstruktur eines Unternehmens dar.[916] Delegation wird in dieser Arbeit als Bestandteil der vertikalen Ablauforganisation aufgefasst und deswegen im Folgenden nicht weiter bewertet.[917] Beide Alternativen der Strukturierung einer Ablauforganisation verteilen die Tätigkeiten, die zur Leistungserbringung notwendig sind, auf organisatorische Stellen innerhalb und ggf. auch außerhalb eines Unternehmens.[918] Eine SOA unterstützt eine prozessorientierte Ablauforganisation, da einzelne Tätigkeiten zur Leistungserbringung identisch auf Services einer Unternehmens-IT, die auf der Basis einer SOA beruht, abgebildet werden können. Eine SOA unterstützt auch eine produktorientierte Ablauforganisation, weil die entsprechenden Tätigkeiten zur Leistungserbringung durch die Unterstützung verteilter Systeme[919] und den Einsatz einer Kommunikationsinfrastruktur[920] unternehmensweit einheitlich allen organisatorischen Stellen angeboten werden können. Infolgedessen können die

---

914 Gaitanides /Prozeßorganisation/ zitiert nach Mellis /Projektmanagement/ S. 72.

915 Vgl. Macharzina /Unternehmensführung/ S. 352; Mellis /Projektmanagement/ S. 80.

916 Vgl. Mellis /Projektmanagement/ S. 80.

917 Wie weiter oben dargestellt, stellen Strukturen vertikaler Ablauforganisation funktionale Anforderungen gegenüber einer Unternehmens-IT dar. Auf diese hat eine Softwarearchitektur keinen Einfluss. Vgl. Kapitel 3.1.3.2.1.

918 Vgl. Mellis /Projektmanagement/ S. 79-80. Mellis geht nicht auf die Alternative der Aufgabenverteilung außerhalb eines Unternehmens ein. Diese kann jedoch als weitere Perspektive ergänzt werden, ohne dass sich ein Einfluss auf die Argumentation ergibt.

919 Vgl. Kapitel 3.2.3.1.1.

920 Vgl. Kapitel 2.2.4.3.

organisatorischen Stellen die Dienstleistungen der Unternehmens-IT entsprechend ihres Zuständigkeitsbereichs (in diesem Fall somit für ein Produkt oder für eine Dienstleistung) nutzen.

Eine Softwarearchitektur folgt dem Prinzip der strukturellen Anatomie dann, wenn es die jeweilige organisatorische Gestaltung eines Unternehmens abbilden kann. Eine SOA ist, wie gezeigt wurde, in der Lage, alle Gestaltungsalternativen – bis auf die vertikale Arbeitsteilung und die Koordination durch persönliche Weisung und Selbstabstimmung – zu unterstützen. Weil sowohl die vertikale Arbeitsteilung als auch die Koordination durch persönliche Weisung und Selbstabstimmung als funktionale Anforderungen an eine Unternehmens-IT formuliert werden müssen, kann davon ausgegangen werden, dass sich die Anatomie einer Unternehmens-IT strukturell an die der Aufbau- und Ablauforganisation eines Unternehmens anpassen kann. Wenn ein Unternehmen als verteilt operierende Einheit strukturiert ist und stärker auf Koordination durch Pläne und Programme als auf Koordination durch persönliche Weisung und Selbstabstimmung aufbaut und die dessen elementaren Einheiten so autonom wie möglich (unabhängig von der Art der Ablauforganisation) arbeiten sollen, folgt eine SOA sogar inhärent dem Prinzip der strukturellen Anatomie, weil die entsprechenden Strukturen den Gestaltungsprinzipien einer SOA folgen.

Autopoiesis

Das Konzept eines offenen, evolutionären und zumindest teilweise autonomen Systems wird autopoietisches System genannt.[921] Anforderungen zur Architekturbewertung, die sich nach Gronau daraus ergeben, sind partielle Autonomie und Selbstorganisation.[922]

Das Prinzip der *partiellen Autonomie* fordert, dass organisatorischen Stellen Dienstleistungen durch die Unternehmens-IT angeboten werden können, damit die organisatorischen Stellen ein Maximum ihrer Aufgaben autonom durchführen können. [923] Das Prinzip bezieht sich auf den festgelegten Aufgabenumfang einer entsprechenden organisatorischen Stelle und muss unabhängig von der organisatorischen Struktur des Unternehmens betrachtet werden, weil sich durch die unterschiedlichen Gestaltungsal-

---

[921] Vgl. Gronau /Wandlungsfähige Informationssystemarchitekturen/ S.219.

[922] Vgl. zum folgenden Absatz Gronau /Wandlungsfähige Informationssystemarchitekturen/ S.219-221.

[923] Vgl. Gronau /Wandlungsfähige Informationssystemarchitekturen/ S. 219.

ternativen der Strukturierung (sowohl der Aufbau- als auch der der Ablauforganisation) entsprechende Abhängigkeiten ergeben. Eine Unternehmens-IT auf Basis einer SOA kann die Tätigkeiten der organisatorischen Stellen exakt durch Services abbilden. Dadurch unterstützt sie die Leistungserbringung der organisatorischen Stellen und erfüllt implizit das Prinzip der partiellen Autonomie. Die Begründung ist: Die in Services abgebildeten Tätigkeiten unterstützen exakt den entsprechenden Arbeitsablauf, und zwar zu den gewünschten Zeitpunkten und für genau die Tätigkeiten, die notwendig sind. Darüber hinaus jedoch können weitere organisatorische Stellen oder Tätigkeiten einbezogen werden, die zu einem entsprechenden Zeitpunkt der Leistungserbringung notwendig sind. Unbedingt zu beachten ist, dass Abläufe innerhalb eines Unternehmens, die eine Zusammenarbeit unterschiedlicher Stellen ad hoc erfordern, nicht explizit durch eine SOA unterstützt werden, z. B. wenn unvorhergesehene Probleme oder Aufgaben bewältigt werden müssen. Dienstleistungen der Unternehmens-IT können jedoch durch eine Orchestrierung von Services so konfiguriert werden, dass diese bezüglich (bis dahin) unvorhergesehenen Aufgaben eingesetzt werden können.[924]

Selbstorganisation bedeutet, dass sich eine Unternehmens-IT und ihre Teile selbstständig verwalten können.[925] Eine Unternehmens-IT erreicht ein hohes Maß an Selbstorganisation dann, wenn sie in der Lage ist, ihre innere Struktur ganz oder teilweise selbst zu bestimmen.[926] Die Selbstorganisation einer Unternehmens-IT wird durch eine SOA unterstützt, weil sie durch den Einsatz der Kommunikationsinfrastruktur fähig ist, ihre innere Struktur – beispielsweise den physischen Ausführungsort eines Services – unabhängig von anderen Komponenten oder den Orten, an denen die Dienstleistung nachge-

---

[924] An dieser Stelle sei jedoch ergänzt, dass eine ad hoc Orchestrierung von Services nicht in der (kurzen) Zeitspanne realisiert werden kann, damit entsprechende Aufgaben zur Lösung von kurzfristig auftretenden Anforderungen an die Unternehmens-IT direkt durch die Unternehmens-IT erfüllt werden können. Dies begründet sich dadurch, da zumindest in den meisten Fällen der Eingriff durch die Softwareentwickler einer Unternehmens-IT notwendig ist, um eine neue Orchestrierung vornehmen zu können. Existieren Lösungen, sodass Anwender eine Unternehmens-IT je nach Anforderung eigenständig Orchestrieren können, kann jedoch auch eine entsprechend kurzfristige Umsetzung erfolgen. Diese Aspekte stellen jedoch technische (und somit nicht beachtete) Anforderungen dar. (Z. B. auch, dass entsprechendes technisches Know-how auf Seiten der Anwender vorhanden ist.)

[925] Vgl. Gronau /Wandlungsfähige Informationssystemarchitekturen/ S.220.

[926] Vgl. Gronau /Wandlungsfähige Informationssystemarchitekturen/ S. 219. Gerade für die organisatorische Einheit, die für die Unternehmens-IT verantwortlich ist (die IT-Abteilung), ist das Kriterium der partiellen Autonomie im Gesamtkontext eines Unternehmens und für die Bewertung dieser Arbeit interessant. Dies liegt daran, da eine Softwarearchitektur direkten Einfluss auf die Aufgaben und Tätigkeiten der IT-Abteilung hat und dadurch den Grad der partiellen Autonomie der IT-Abteilung mitbestimmen kann.

fragt wird,[927] – automatisch (und eigenständig) gestalten zu können. Die Kommunikationsinfrastruktur einer SOA ermöglicht einen dynamischen Aufruf von Services durch Dienstleistungsnehmer. Die Unternehmens-IT wird dadurch in die Lage versetzt, die Ausprägungen der inneren Struktur im laufenden Betrieb ändern zu können.

3.2.3.2.2 Nutzen der Nachhaltigkeit

Das Kriterium der dauerhaften Zukunftsfähigkeit einer Unternehmens-IT kann über strukturanaloge und autopoietische Merkmale hinaus anhand der technisch-fachlichen Qualitätsattribute beurteilt werden. Diese können in entsprechenden Ausprägungen zu einer höheren Nachhaltigkeit einer Unternehmens-IT führen als durch jeweils andere Ausprägungen. Einfluss auf die Nachhaltigkeit haben die technisch-fachlichen Qualitätsattribute, die durch ihre Eigenschaften bedingt eine Wirkung in der Zukunft haben. Hierunter fallen Wandlungsfähigkeit, Wiederverwendbarkeit, Portabilität und Effizienz. Die Wandlungsfähigkeit ist hierbei von großer Bedeutung, weil Wandlungsbedarf über den gesamten Lebenszyklus einer Unternehmens-IT eintreten kann.[928] Eine hohe Fähigkeit zur Wandlung sorgt dafür, dass eine Unternehmens-IT auch dauerhaft zukunftsfähig sein kann, weil sie sich neuen Anforderungen entsprechend wandeln kann. Eine Unternehmens-IT besitzt eine hohe Zukunftsfähigkeit, wenn eine hohe Wiederverwendbarkeit der Unternehmens-IT und insbesondere ihrer Teile vorliegt. Dadurch besteht die Möglichkeit, existierende und durch die Unternehmens-IT unterstützte (Teil-)Aufgaben und (Teil-)Tätigkeiten in neuen und veränderten Geschäftsprozessen zu verwenden. Neue Technologien und Produkte zur Mobilisierung der Arbeitnehmer werden vermehrt eingesetzt und können die Produktivität der Mitarbeiter weiter erhöhen. In diesen technisch-fachlichen Qualitätsattributen wurde eine SOA positiv bewertet.[929]

Um im gesamten Lebenszyklus einer Unternehmens-IT auf solche Entwicklungen reagieren zu können, unterstützt hohe Portabilität – das dritte Kriterium – die Zukunftsfä-

---

[927] Vgl. Kapitel 3.2.2.7.2.

[928] Moro und Lehner beispielsweise nennen in ihrem Fallbeispiel die ‚Skalierbarkeit' (vgl. Moro, Lehner /Reengineering/ S. 29). Diese ist eine Anforderung, die die zukünftige Notwenigkeit eine Unternehmens-IT in ihrer Verarbeitungskapazität erweitern werden zu müssen, bedeutet. Eine skalierbare Unternehmens-IT beispielsweise ist dadurch dauerhaft zukunftsfähig, da ein (starkes) Wachstum der Verarbeitungsansprüche in Zukunft nicht zu einer kompletten Neuentwicklung einer Unternehmens-IT führen wird; zudem können die Ansprüche der Nutzer der Unternehmens-IT erfüllt werden.

[929] Vgl. zur Wandlungsfähigkeit Kapitel 3.2.2.2 und zur Wiederverwendbarkeit Kapitel 3.2.2.6.

higkeit einer Unternehmens-IT; durch sie ist eine Unternehmens-IT auf unterschiedlichen technischen Grundlagen funktionsfähig. Effizienz als viertes Kriterium zur Bewertung der Nachhaltigkeit deutet an, wie hoch ein aktueller und zukünftiger Ressourceneinsatz sein muss, um eine Unternehmens-IT betreiben zu können und welches Antwortzeitverhalten zukünftig zu erwarten ist. Das Antwortzeitverhalten zeigt z. B. wie hoch ein Verarbeitungspotential einer Unternehmens-IT ist und sein wird. Gutes Antwortzeitverhalten – also die Ausprägung, korrekte Antworten in kurzer Zeit liefern zu können – unterstützt die Nachhaltigkeit, weil eine Unternehmens-IT auch höhere zu erwartende Daten- und Verarbeitungsvolumen verarbeiten kann. Portabilität und Effizienz einer SOA wurden negativ bewertet.[930]

Die technisch-fachlichen Qualitätsattribute Funktionserfüllung, Benutzbarkeit und Verlässlichkeit haben keinen Effekt auf zukünftige Entwicklungen, die über Gegenwartsaspekte hinausgehen. Beispielsweise hat die Funktionserfüllung zum Zeitpunkt der Inbetriebnahme einer Unternehmens-IT den gleichen Stellenwert wie zu jedem beliebigen Zeitpunkt nach der Inbetriebnahme, wohingegen die oben genannten Qualitätsattribute zu zukünftigen Zeitpunkten weiteres Gewicht erlangen. Aus diesem Grund findet bei der Bewertung technisch-fachlicher Qualitätsattribute auch keine Doppelbewertung dieser Qualitätsattribute statt, weil hier explizit die Entwicklung über den Lebenszyklus einer Unternehmens-IT betrachtet wird. Die Benutzbarkeit und Verlässlichkeit haben der Funktionserfüllung entsprechend ebenso keine zusätzliche zukünftige Bedeutung. Deshalb werden diese Qualitätsattribute nicht in die Bewertung der Nachhaltigkeit aus technisch-fachlichen Kriterien aufgenommen.

Die Effizienz wurde negativ bewertet, jedoch erhält die Nachhaltigkeit durch technisch-fachliche Kriterien eine positive Bewertung, weil sowohl die Bewertung der Wandlungsfähigkeit und Wiederverwendbarkeit als auch die Bewertung der Portabilität positiv ausgefallen ist.

3.2.3.2.3 Gesamtbewertung der Nachhaltigkeit

Zusammenfassend wird das Qualitätsattribut ‚Nachhaltigkeit' positiv bewertet (Tab. 3-19).

---

[930] Vgl. zur Portabilität Kapitel 3.2.2.7 und zur Effizienz Kapitel 3.2.2.5.

(Unter-)Qualitätsattribute	*Wirkung*
Technisch-fachliche Kriterien	+
Strukturanalogie	+
Autopoiesis	+
Gesamtbewertung	+

Tab. 3-19: Nutzenbewertung des Qualitätsattributes Nachhaltigkeit

3.2.3.3 Integriertheit

Integriertheit einer Softwarearchitektur ist die Fähigkeit, Mechanismen der Konnektivität und Interoperabilität anbieten zu können, damit dem Ökosystem eines Unternehmens Dienstleistungen, Werkzeuge und Technologien zur Verfügung gestellt werden können mit dem Ziel, das Ökosystem zu erhalten, die Leistungsfähigkeit des Ökosystems zu verbessern und an das Unternehmen gestellte Informationsansprüche anforderungsgerecht bedienen zu können.[931]

Im Folgenden werden auf Basis von Argumentationen Bewertungen der Unterqualitätsattribute der Integriertheit vorgenommen. Anschließend werden die Wirkungen einer SOA auf die Integriertheit einer Unternehmens-IT untersucht und es wird eine Gesamtbewertung aufgestellt.

3.2.3.3.1 Nutzen der Unterqualitätsattribute der Integriertheit

Die Unterqualitätsattribute der Integriertheit sind Integrationsgegenstand, Integrationsreichweite und Integrationsrichtung.[932]

[931] Vgl. Kapitel 3.1.4.4.2.

[932] Vgl. Kapitel 3.1.4.4.2.

Integrationsgegenstand

Zur Bewertung des Integrationsgegenstandes wird die Fähigkeit einer Software untersucht, Daten, Funktionen und Programme verschiedener Unternehmensbereiche und Unternehmensebenen integrieren zu können.[933]

Eine SOA unterstützt die Integration von Funktionen und Programmen. Funktionen werden in einer SOA in Services abgebildet und können durch die Verwendung einer Kommunikationsinfrastruktur von anderen Teilen der Unternehmens-IT (selbst wenn diese örtlich verteilt aufgebaut ist) genutzt und dadurch integriert werden. Programme können auch durch eine SOA über einen Umweg integrationsfähig gestaltet werden. Es werden Services für Schnittstellen von Programmen oder um gesamte Programme herum gebildet, die wiederum die Dienstleistungen der Schnittstellen bzw. der Programme anderen Teilen einer Unternehmens-IT über die Kommunikationsinfrastruktur einer SOA zur Verfügung stellen können.[934]

Die Integration von Daten kann durch eine SOA nicht erreicht werden, weil Services nur Funktionen als Dienstleistungen anbieten können. Als Schnittstelle zu Daten verarbeitenden Stellen (z. B. Services) könnten Dienstleistungen erstellt werden, die Daten in bestimmter Form oder in bestimmter Kombination bereitstellt, oder die das Datenmanagement für bestimmte Datenbanken und Teile von (verschiedenen) Datenbanken übernimmt. Eine Integration auf Datenebene wird dadurch jedoch nicht erreicht, sondern vielmehr eine integrierte Datenrepräsentation.

Betrachtet man die potentiellen Integrationsgegenstände, so kann Integration (als Vorgang[935]) danach beurteilt werden, wie bereits vorhandene Teile einer Unternehmens-IT zusammengefügt oder neue Teile in eine vorhandene Struktur eingebunden werden

---

933 Vgl. Kapitel 3.1.4.4.2.

934 Solche Services werden in Folgenden ‚Wrapper' genannt. Vgl. Aier, Schönherr /Flexibilisierung/ S. 37-38; Ließmann, Wetzke /Integrationsinfrastruktur/ S. 205-206.

935 Vgl. zur Integration als Zustand und Integration als Vorgang Linß /Nutzeffekte/ S. 5-7. In dieser Arbeit wird nicht explizit zwischen der Integration als Zustand und der Integration als Vorgang unterschieden. Dem Verständnis des Autors dieser Arbeit nach muss jeder Integration als Zustand eine Integration als Vorgang vorhergegangen sein. Eine Unterscheidung in diese Begrifflichkeiten erscheint für die Bewertung der Integriertheit einer Unternehmens-IT jedoch nicht hilfreich, da die Integriertheit den Zustand einer Unternehmens-IT bewertet, dem somit immer auch eine Integration als Vorgang vorausgegangen sein muss (implizit bei einer Neuentwicklung, explizit bei einer Umstellung vorhandener Teile einer Unternehmens-IT). Die Bewertung in dieser Arbeit umfasst somit implizit beide Unterscheidungsansätze.

können.[936] Die Integration der integrationsfähigen Integrationsgegenstände (Funktionen und Programme) durch eine SOA kann sowohl bei bereits bestehenden als auch bei neuen Teilen einer Unternehmens-IT durch die Verwendung der Kommunikationsinfrastruktur unproblematisch geschehen. Dies begründet sich dadurch, weil keine speziellen Schnittstellen zu allen bereits bestehenden Teilen einer Unternehmens-IT einzeln zur Verfügung gestellt werden müssen, sondern nur eine Schnittstelle zur Kommunikationsinfrastruktur erforderlich ist, damit der zu integrierende Teil einer Unternehmens-IT integriert werden kann.[937] Im Falle schon bestehender Teile einer Unternehmens-IT, z. B. bestehender Programme, ist eine Integration schwieriger als bei neu zu integrierenden Teilen, weil diese speziell auf Integrationsaspekte hin optimiert werden kann. Hierunter fallen z. B. einheitliche Datenformate, Schnittstellenparameter und Ähnliches.

Integrationsreichweite

Zur Bewertung der Integrationsreichweite wird die Fähigkeit einer Software untersucht, eine innerbetriebliche als auch zwischenbetriebliche Integration unter Einbeziehung verschiedener Unternehmensbereiche und -ebenen erreichen zu können.[938]

Eine Unternehmens-IT auf Basis einer SOA kann potentiell alle unternehmensinternen Unternehmensbereiche und Unternehmensebenen integrieren. Der Grund ist, weil die Funktionen und Programme einzelner Unternehmensbereiche integrationsfähig sind und innerhalb des Unternehmens eine entsprechende Kommunikationsinfrastruktur existiert, die das gesamte Unternehmen abdecken kann.

Unternehmensübergreifend ist eine Integration auch möglich, wenn externen Integrationsgegenständen eine entsprechende Kommunikationsschnittstelle zur Kommunikationsinfrastruktur angeboten wird. Über diese Kommunikationsschnittstelle kann auf die vorgesehenen und freigegebenen Bestandteile einer Unternehmens-IT zugegriffen bzw. kommuniziert. Entsprechende Sicherheitsmechanismen können hierbei von der Kommunikationsinfrastruktur wahrgenommen werden.

---

[936] Vgl. Linß /Nutzeffekte/ S. 6.

[937] Vgl. Aier, Schönherr /Flexibilisierung/ S. 14.

[938] Vgl. Kapitel 3.1.4.4.2.

Integrationsrichtung

Zur Bewertung der Integrationsrichtung wird die Fähigkeit einer Software untersucht, horizontale und vertikale Integration zu unterstützen.[939]

Horizontale Integration einer Unternehmens-IT betrifft die Integration von (auch externen) Bereichen eines (oder mehrerer) Unternehmen. Sie kann nach der Anzahl der integrierten Bereiche und nach der Enge der Integration bewertet werden.[940] Wie bei der Bewertung der Integrationsgegenstände gezeigt, lassen sich alle Bereiche einer Unternehmens-IT durch eine SOA integrieren. Die ‚Enge' der Integration wird nach Linß in drei Kategorien unterschieden. Die höchste Kategorie ist die, dass Tätigkeiten über die gesamte Wertschöpfungskette autonom vom System angestoßen werden können.[941] Können in einer Unternehmens-IT, die auf Basis einer SOA aufgebaut wird, Funktionen als Services aufgebaut werden, und müssen nur wenige Integrationsgegenstände über Wrapper eingebunden werden, ist eine nach dieser Definition ‚enge' Integration erreichbar, weil entsprechende Dienstleistungen der Unternehmens-IT in der notwendigen Reihenfolge angestoßen werden können. Müssen dagegen ganze Programme über Adapter angebunden werden, ist ein automatisches Anstoßen aller notwendigen Dienstleistungen nicht sichergestellt, weil diese unter Umständen von internen Zusammenhängen eines Programms am automatischen Ablaufen gehindert werden. Beispielsweise kann das zusätzliche Anstoßen eines Programmteils durch einen Anwender oder das Festlegen bestimmter Ablaufparameter – wie Speicherverzeichnisse innerhalb eines Programms – noch notwendig werden. Die zweithöchste Bewertung der ‚Enge' (auf einer dreistufigen Skala) ist eine gemeinsame Datenbasis und aufeinander abgestimmte Funktionen und Programme. Eine gemeinsame Datenbasis gegenüber Funktionen und Programmen kann in einer SOA durch die gemeinsame Datenpräsentation[942] und eine unternehmensweit einheitliche Datenherkunft erreicht werden. Eine einheitliche Datenherkunft wird erreicht, weil Daten unternehmensweit und über Unternehmensgrenzen hinweg verfügbar sein können.[943] Die Funktionen und Programme sind durch den Einsatz von Services und der Kommunikationsinfrastruktur aufeinander abgestimmt. Die

---

[939] Vgl. Kapitel 3.1.4.4.2.

[940] Vgl. Linß /Nutzeffekte/ S. 23.

[941] Vgl. Linß /Nutzeffekte/ S. 24.

[942] Vgl. Kapitel 3.2.2.1.1.

zweite Stufe der ‚Enge' nach Linß kann somit durch eine Unternehmens-IT auf Basis einer SOA erreicht werden.

Vertikale Integration einer Unternehmens-IT betrifft die Integration von Daten und Prozessen zur Leistungserstellung eines Unternehmens in Administrations- und Kontrollaufgaben. Die nach Linß höchste Stufe der vertikalen Integration wird dann erreicht, wenn eine hohe Datenverdichtung auf Planungs- und Kontrollebene möglich ist.[944] Linß versteht darunter die Fähigkeit einer Unternehmens-IT, den Unternehmensebenen, die Führungs- und Kontrollaufgaben wahrnehmen müssen, Echtzeitdienstleistungen[945] untergeordneter administrativer sowie produktiver Systeme zur Verfügung stellen zu können.[946] Linß stellt diesen datenbasierten Bewertungsansatz auf, weil zur Administration und Kontrolle primär Daten und keine Funktionen untergeordneter Bereiche notwendig sind. Legt man diesen Bewertungsansatz zugrunde unterstützt eine SOA eine vertikale Integration nicht, weil die Integration von Daten nicht erreicht werden kann. Jedoch kann durch Funktionen zur Datenlieferung zum Zwecke der Administration und Kontrolle, die wiederum als Services in der Unternehmens-IT angeboten werden, zumindest der Anspruch erfüllt werden, dass Echtzeitdaten geliefert werden können. Beachtet man andererseits, dass zu administrativen Aufgaben und Kontrollaufgaben auch eine Steuerung und Leitung von Aktivitäten untergeordneter Stellen notwendig ist und dies über die Nutzung von existierenden Funktionen untergeordneter Ebenen möglich sein kann, muss der rein datenorientierte Bewertungsansatz von Linß um die Bewertung der Steuerungsmöglichkeit durch Prozesseingriffe erweitert werden. Solche Eingriffe in Prozesse und Planungen können am besten an den Stellen erfolgen, die entsprechende Prozesse steuern bzw. Planungen aufstellen – also typischerweise die Funktionen, die auf untergeordneten Ebenen alltäglich durchgeführt werden. Kann die Leitungs- und Kontrollebene auf diese Funktionen zugreifen und entsprechende Änderungen durchführen, wird eine hohe vertikale Integration erreicht, die nicht nur berichtend an Leitungs- und Kon-

---

[943] Vgl. Kapitel 3.2.2.1.2.

[944] Vgl. Linß /Nutzeffekte/ S. 24.

[945] Vgl. Keller /EAI/ S. 26-27. Unter Echtzeitdienstleistungen fallen Dienstleistungen bezüglich Prozessen und Daten. Aus Sicht eines Dienstleistungsnehmers wird dies ‚Echtzeitanforderung' genannt. Vgl. Mellis /Projektmanagement/ S. 5. Für ein Beispiel vgl. Dostal, Jeckle, Kriechbaum /Semantik/ S. 46-47.

[946] Vgl. Linß /Nutzeffekte/ S. 24; Keller /EAI/ S. 45-46. Keller erweitert diese Sicht um die Möglichkeit der Priorisierung von Datenanfragen, wodurch ‚Echtzeitdaten' nach Keller kategorisiert und unterschiedliche Qualitätsanforderungen an den ‚Echtzeitcharakter' realisiert werden können.

trollstellen, sondern auch umgekehrt steuernd auf untergeordnete administrative und produktive Stellen wirken kann. Eine Unternehmens-IT auf Basis einer SOA kann dies dadurch erreichen, weil die Funktionen untergeordneter administrativer und produktiver Stellen über die Kommunikationsinfrastruktur auch durch die übergeordneten administrativen und kontrollierenden Stellen ansprechbar sind.

3.2.3.3.2 Nutzen der Integriertheit

Integration muss auch unter Beachtung der zu unterstützenden Entwicklungsprozesse bewertet werden.[947] Der Entwicklungsprozess kann nach gängigen Vorgehensmodellen wie z. B. dem Wasserfallmodell oder iterativen Vorgehensweisen organisiert sein.[948] Das Konzept SOA bietet keine Unterstützung eines spezifischen Entwicklungsprozesses, auch wird die Erstellung einer SOA nicht explizit nur durch ein etabliertes Entwicklungsvorgehen unterstützt. Weil eine SOA jedoch auf Konzepte der Komponentenorientierung zurückgreift, können entsprechende Vorgehensweisen adaptiert werden und als Unterstützung des Entwicklungsprozesses dienen. Die Unterstützung des Entwicklungsprozesses ist dennoch (noch) nicht durch etablierte Vorgehensweisen unterstützt und wird deswegen negativ bewertet.

3.2.3.3.3 Gesamtbewertung der Integriertheit

Zusammenfassend wird das Qualitätsattribut ‚Integriertheit' positiv bewertet (Tab. 3-20).

(Unter-)Qualitätsattribute	*Bewertung*
Entwicklungsprozess	–
Integrationsgegenstand	+
Integrationsreichweite	+
Integrationsrichtung	+

[947] Vgl. Zahavi /Integration/ S. 25-40.

[948] Vgl. Zahavi /Integration/ S. 25-40.

Gesamtbewertung	+

Tab. 3-20: Nutzenbewertung des Qualitätsattributes Integriertheit

3.2.4 Normalisierung der Nutzenpotentiale

Eine Normalisierung der bisherigen Ergebnisse wird nun durch die Gewichtungen, die bei der Aufstellung des Qualitätsmodells identifiziert wurden, vorgenommen.[949] Positive Bewertungen erfahren durch eine hohe Gewichtung eine weitere Verstärkung und neutrale behalten ihre Ausprägung. Um die Abstufungen nicht zu fein werden zu lassen, wird ein geringes Gewicht wie ein neutrales behandelt.[950] Neutrale Bewertungen können durch Gewichte nicht weiter verändert werden. Die daraus resultierende Transformation ist in Abb. 3-8 dargestellt.

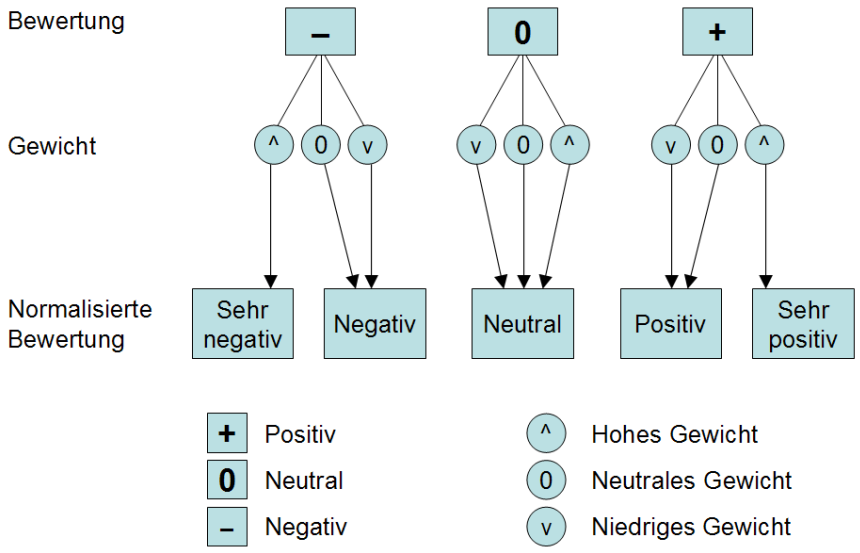

Abb. 3-8: Normalisierung

[949] Andere Bewertungsvorgehen folgen einer vergleichbaren Bewertungsmethode. Z. B. verwendet die CBAM Gewichtungen, um zu Nutzenwerten zu gelangen; vgl. Asundi, Kazman, Klein /Economic Approach/ S. 14. Im QFD wird an mehreren Stellen von Gewichtungen Gebrauch gemacht, um zu einem detaillierten Ergebnis zu gelangen vgl. Herzwurm, Schockert, Mellis /Qualitätssoftware/ S. 40-41, S. 73-76 & S. 93-97.

[950] Dadurch wird zwar keine äquidistant korrekte Normalisierung erreicht, für die Verständlichkeit und die Aussagekraft der Ergebnisse dieser Arbeit ist ein solches Vorgehen jedoch praktikabel und unkritisch, weil eine weitere Abstufung in z. B. positiv, sehr positiv und extrem positiv qualitativ keinen Zusatznutzen bringen würde. Das Vorgehen entspricht desweiteren den Eigenschaftne einer Or-

In der Praxis müssen die hier getroffenen Bewertungen durch Ergänzungen bezüglich der gewählten Technologien ergänzt werden. Auch kann eine Anpassung der Gewichtungen erfolgen, um Anforderungen des konkreten Falls genauer abbilden zu können. Die Ergebnisse der Normalisierung sind in Tab. 3-21 dargestellt.

Qualitätsattribut	*Ausprägung*	*Gewichtung*	*Bewertung*
Funktionserfüllung	+	v	Positiv
Wandlungsfähigkeit	+	^	Sehr positiv
Benutzbarkeit	+	^	Sehr positiv
Verlässlichkeit	+	^	Sehr positiv
Effizienz	–	v	Negativ
Wiederverwendbarkeit	+	^	Sehr positiv
Portabilität	–	^	Sehr negativ
Strategieunterstützung	+	^	Sehr positiv
Nachhaltigkeit	+	^	Sehr positiv
Integriertheit	+	^	Sehr positiv

Tab. 3-21: Bewertung der gewichteten Qualitätsattribute[951]

3.2.5 Nutzenpotential durch die Verwendung von Standards

In der Literatur finden sich vielfältige Beiträge zur Verwendung von Standards in der Softwareentwicklung bzw. Beiträge, die die Verwendung von Standards ansprechen.[952] Auch bei der Diskussion über SOA wird die Diskussion um Standards häufig einbezo-

dinalskals, bei der über Differenzen zwischen den Merkmalsausprägungen keine Aussagen gemacht werden. Vgl. Harms /Statistik/ S. 16-18.

[951] Zur Legende vgl. Abb. 3-8.

[952] Vgl. beispielsweise Aier, Schönherr /Flexibilisierung/ S. 14; Dostal, Jeckle /Service-orientierte Architektur/ S. 54; Raasch /Systementwicklung/ S. 36.

gen und als Argument für SOA herangezogen.[953] Deshalb wird an dieser Stelle die Bedeutung von Standards kurz erläutert und in Zusammenhang zu Softwarearchitekturen gesetzt. In dieser Arbeit soll verständlich gemacht werden, ob und falls ja, welche Vorteile durch den Einsatz von Standards in einer Unternehmens-IT, die auf Basis einer SOA aufgebaut ist, existieren. Als Ziele für den Einsatz von Standards werden dabei Plattform- und Sprachunabhängigkeit, Kostensenkung, Effizienzsteigerung der Prozesse und Erhöhung der Betriebssicherheit genannt.[954]

Statt Ziele für den Einsatz aufzuzeigen wird daneben vereinzelt argumentiert, welchen Nutzen der Einsatz von Standards tatsächlich bieten kann. Beispielsweise dient eine Verwendung von Standards der Erhöhung der Portabilität[955] und der Interoperabilität[956], und es kann eine einfache Interaktion verschiedener Systemkomponenten erreicht werden.[957] Argumente aus der Organisationsforschung bezüglich der Nutzung von Standards lassen sich auch auf den Einsatz von Standards in einer Unternehmens-IT übertragen: Sie ermöglicht durch ihren Charakter als Instrument zur Vorauskoordination den Verzicht auf weitere Koordinationsinstrumente.[958] Die Diskussion um Standards kann auch aus der Sicht des Produktmarketings auf eine Unternehmens-IT angewendet werden. Demzufolge sind Standards ein wesentliches Instrument im Kampf um Kunden und Marktanteile. Durch den Einsatz von Standards lassen sich Netzwerkeffekte[959], eine starke Kundenbindung (Linkage)[960] und Lock-In von Kunden[961] erreichen. Betrachtet man eine Unternehmens-IT als Dienstleister für ein Unternehmen können diese Effekte auch auf ein Unternehmen als Kunden der Unternehmens-IT übertragen werden.

---

953 Vgl. Karch u. a. /SAP/ S. 33-36. Karch bezieht sich bei seiner Argumentation auf die Enterprise Service Architecture (ESA) von SAP, die er als Umsetzung einer SOA vorstellt.

954 Vgl. Wehle /Business Continuity/ S. 226; Schuppe, Cassens /IT im Handelsraum/ S. 401-402; Dostal, Jeckle /Service-orientierte Architektur/ S. 54-55.

955 Vgl. Dustdar, Gall, Hauswirth /Software-Architekturen/ S. 72; Bass u. a. /Software architecture/ S. 94.

956 Vgl. Bass u. a. /Software architecture/ 94.

957 Vgl. zu diesem Absatz Willgosch /Informationslieferanten/ S. 308-311; König, Weitzel, Martin /Straight Through Processing/ S. 409-411.

958 Vgl. Kieser, Walgenbach /Organisation/ S. 136. Kieser und Walgenbach betrachten diesen Zusammenhang bei Rollenstandardisierung in Organisationen. Die Argumentation lässt sich jedoch auf Softwarekomponenten und Schnittstellen übertragen, da sich das Zusammenwirken von Softwarekomponenten auf das Zusammenarbeiten von Mitarbeitern in Unternehmen übertragen lässt.

959 Karch u. a. /SAP/ S. 35-36.

960 Vgl. Laudon, Laudon /Business/ S. 55-59.

961 Vgl. Laudon, Laudon /Business/ S. 55-59.

Die Kompatibilität der Bestandteile einer Unternehmens-IT untereinander und zu externen Komponenten ist, vor allem wegen der teilweise langen Betriebszeiten einer Unternehmens-IT, sowohl ein wichtiges Bewertungs- bzw. Entscheidungskriterium als auch Ziel.[962] Unterschieden werden kann hier zwischen einer rückwärtigen und aufwärtigen Kompatibilität.[963] Besonders in Bezug auf Kompatibilität können Standards Unterstützung geben. Greifen z. B. alle Bestandteile einer Unternehmens-IT bezüglich Kommunikationsprotokollen, Kommunikationsformaten und Speicherformaten auf Standards zurück, können neue Bestandteile durch die Unterstützung derselben Standards kompatibel zur bestehenden Unternehmens-IT sein. Dadurch werden Sie ohne Zusatzaufwand und ohne potentielle Kommunikationsprobleme in einer Unternehmens-IT eingesetzt.

Tatsächlich lassen sich einige Argumente auf Gestaltungsprinzipien einer SOA übertragen: Einheitliche Schnittstellen der Services können als Zugriffsstandard für einen Service interpretiert werden, über den jeder Dienstleistungsnachfrager die entsprechende Dienstleistung abrufen kann. Durch die Verwendung etablierter Standards für die Kommunikationsprotokolle ist es weiterhin möglich, Services interoperabel zu erstellen. Es ist dadurch möglich, Services von externen, nicht explizit dazu vorgesehenen Kommunikationspartnern ansprechen zu lassen.[964]

Grundsätzlich jedoch können Standards unabhängig von einer jeweiligen zugrunde gelegten Softwarearchitektur verwendet werden. Es besteht eine begrenzte Wahlmöglichkeit, wenn technische Entscheidungen bestimmte Gestaltungsalternativen bezüglich Standards einschränken. Z. B. kann gegen den Einsatz von XML als Kommunikationsformat sprechen, dass bestimmte Bestandteile einer Unternehmens-IT XML nicht beherrschen und eine Umstellung nicht möglich ist.[965] Im Zusammenhang mit SOA werden Standards oftmals hervorgehoben, weil der Architekturansatz keine technologischen Vorschriften formuliert und somit eine uneingeschränkte Wahlmöglichkeit besteht.[966] Dies erklärt den Grund für die häufige Diskussion von Standards im Zusammenhang mit SOA.

---

[962] Vgl. Moro, Lehner /Reengineering/ S. 28.

[963] Vgl. zu Definitionen beispielsweise IEEE /Glossary/ S. 84 und IEEE /Glossary/ S. 26.

[964] Vgl. Moro, Lehner /Reengineering/ S. 48-49.

[965] Darüber hinaus finden sich auch Hinweise darauf, dass der Einsatz von XML in der Praxis tendenziell mit einer Erhöhung des zu übermittelnden Datenvolumens einhergeht. Vgl. Chappell /ESB/ S. 113.

Zusammenfassend kann festgehalten werden, dass die Diskussion um den Einsatz von Standards durch ihre potentiell positiven Auswirkungen gerechtfertigt ist. Jedoch lässt sich im Zusammenhang mit Softwarearchitekturen – konkrete technische Realisierungen außer Acht gelassen – keine signifikante Entscheidungshilfe bzw. kein Nutzen identifizieren, der einer spezifischen Softwarearchitektur, inklusive einer SOA, zugerechnet werden kann.

---

[966] In der Praxis wird man jedoch durch die Wahl beispielsweise der Kommunikationplattform Einschränkungen erfahren.

4. Kostenpotential einer SOA

Nach dem Nutzenpotential einer SOA wird als Nächstes das Kostenpotential einer SOA untersucht. Software wird – nach gängiger Meinung – immer umfangreicher, komplexer und teurer.[967] Außerdem weisen zwei Unternehmens-IT-Systeme, die vergleichbaren Anforderungen unterliegen, i. d. R. nicht identische Kosten auf. Demzufolge steuern bestimmte Ausprägungen einer Unternehmens-IT einen spezifischen Beitrag zu ihren Kosten bei. Neuere Entwicklungen und Vorgehensweisen im Bereich der Softwareentwicklung weisen somit das Potential auf, die Kosten zu beeinflussen – mit dem Ziel, diese zu senken. Diese Wirkungen werden im nächsten Kapitel (Kapitel 5) bei der Betrachtung des Wirtschaftlichkeitspotentials eines SOA untersucht, weil Wirtschaftlichkeitseffekte auf die Kombinationen des Nutzenpotentials (Kapitel 3) mit dem Kostenpotential (dieses Kapitel, Kapitel 4) zurückgeführt werden.

Im Folgenden werden zunächst Bewertungsalternativen und –probleme aufgezeigt (Kapitel 4.1). Anschließend werden Kosten identifiziert, die in direktem Zusammenhang mit SOA stehen. Darunter fallen Kosten der Infrastruktur (Kapitel 4.2), Kosten der Entwicklung (Kapitel 4.3), Kosten durch organisatorischen Wandel (Kapitel 4.4), Kosten der Ablösung von Legacy Systemen (Kapitel 4.5) und Kosten des Managements einer SOA (Kapitel 4.6).

4.1 Bewertungsalternativen und -probleme

Kostenbewertungen können auf verschiedenen Begrifflichkeiten von Kosten beruhen. Reifer z. B. unterscheidet Kosten in die Kategorien der wiederkehrenden und nicht-wiederkehrenden Kosten.[968] Nach Reifer sind Beispiele für nicht-wiederkehrende Kosten die Kosten der Hardwarebeschaffung, Kosten für Schulungsmaßnahmen, für neue Software und auch die Kosten der Entwicklung einer Software. Teilweise lassen sich diese Kosten direkt ermitteln (z. B. Kosten für Hardwareanschaffungen). Teilweise existieren verschiedene Bewertungsmodelle, um Kosten ex-ante zu ermitteln, z. B. auch

---

[967] Vgl. Boehm /Economics/ S. 17-18; Ambler /Modeling/ S. 3; Sommerville /Software Engineering/ S. 19-20; Mellis /Projektmanagement/ S. 27.

[968] Vgl. Reifer /Business Case/ S. 61.

für Entwicklungskosten von Software.[969] Wiederkehrende Kosten dagegen sind Kosten, die mehrfach – über den Lebenszyklus einer Software betrachtet – auftreten. Solche Kosten können sich im Laufe der Zeit ändern, z. B. Kosten der Wartung, die zunächst nur sporadisch und in geringem Umfang auftreten.[970] Mit zunehmendem Alter der Unternehmens-IT und mit der zunehmenden Häufigkeit neuer und geänderter Anforderungen steigen die Wartungskosten.[971]

Kosten können begrifflich auch in tangible und intangible Kosten unterschieden werden. Nach Brynjolfsson und Yang sind intangible Kosten solche, die häufig nur unzureichend in quantitative Ausprägungen übertragen werden können (z. B. in monetäre oder zeitliche Skalen).[972] Intangible Kosten können unter Umständen sogar in keiner Weise durch den Kostenverursacher bemerkt werden, z. B. dann, wenn Kosten außerhalb eines Projektes und außerhalb eines Unternehmes entstehen und diese innerhalb des Projekte bzw. innerhalb eines Unternehmens nicht als Kosten identifiziert werden.[973] (Oft werden solche Kosten auch ignoriert, weil das Projektbudget bzw. die Gewinne des eigenen Unternehmens nicht direkt beeinflusst werden.) Tangible Kosten sind dagegen Kosten, die eindeutig identifiziert, zugeordnet und in quantitative Ausprägungen überführt werden können.

Kosten der Informationsverarbeitung lassen sich nur schwer abschätzen, und Ergebnisse werden i. d. R. nicht offen gelegt.[974] Gerade für die vorliegende Arbeit erschwert dies eine umfassende Analyse der spezifischen, mit einer SOA zusammenhängenden Kosten. Die im Zuge dieser Arbeit geführten Interviews bestätigen, dass Firmen die Kosten, die mit der Entwicklung einer SOA zusammenhängen, nicht darlegen wollen.[975] Auch werden Informationen zu fehlgeschlagenen Projekten i. d. R. nicht veröffentlicht, um da-

---

[969] Beispielsweise zählen hierzu die Function Point Methode und das Cocomo (nach Boehm). Vgl. Coombs /Estimation/ S. 33-55; Krüger, Seelmann-Eggebert /IT-Architektur/ S. 180-181; Boehm u. a. /Cocomo II/.

[970] Vgl. Hares, Royle /Measuring/ 231.

[971] Vgl. Hares, Royle /Measuring/ 231.

[972] Vgl. Brynjolfsson, Yang /Productivity/ S. 208.

[973] Vgl. Alshawi, Baldwin, Irani /Benchmarking/ S. 415; Hochstrasser /IT investments/ S. 215; Krüger, Seelmann-Eggebert /IT-Architektur/ S. 111.

[974] Vgl. Moormann /Umbruch/ S. 11.

[975] Vgl. Kapitel 1. Der Autor dieser Arbeit vermutet beispielsweise, dass viele Firmen mit der Einführung einer SOA neue und ungewohnte Vorgehensweisen einführen und deswegen zusätzliche, vermutlich

durch entstehende negative Publizität zu vermeiden. Informationen können deshalb nur für einen Teil aller Projekte – nämlich für erfolgreiche Projekte – gesammelt werden und diese enthalten i. d. R. keine Kostenangaben. Aus diesem Grund muss auch bei der Identifikation von Kosten auf die Philosophie des kritischen Realismus zurückgegriffen werden. Nach dieser ist es teilweise nicht vermeidbar, zu wissenschaftlichen Erkenntnissen durch Kreativität zu gelangen.[976] Leider kann deswegen nicht der Anspruch erhoben werden, dass in diesem Kapitel erschöpfend und vollständig alle Kostenpotentiale identifiziert werden können, die in direktem Zusammenhang mit SOA auftreten.

Kosten haben direkte Auswirkungen auf den Kostenträger und können häufig mit größerer Sicherheit bestimmt werden als Nutzeneffekte.[977] Die Kosten für die Anschaffung einer neuen Maschine z. B. werden direkt dem Budget einer Abteilung zugerechnet, wohingegen ein potentieller Produktionszuwachs zwar geschätzt, jedoch erst im Nachhinein quantitativ beziffert werden kann. Hat die neue Maschine dagegen Auswirkungen auf die Qualität des Produktes, kann unter Umständen nie ein direkter quantitativer Zusammenhang zu Erlösen festgestellt werden.[978] Aus diesem Grund wird in dieser Arbeit davon ausgegangen, dass Kosten direkt ermittelbar sein müssen und tangibel sind.[979] Des Weiteren werden sowohl wiederkehrende als auch nicht-wiederkehrende Kosten identifiziert.

4.2 Kosten der Infrastruktur

Die Infrastruktur einer Unternehmens-IT umfasst Komplementärprodukte, da diese nur dann betrieben werden kann.[980] Zu Komplementärprodukten zählen mindestens die Hardware (z. B. Prozessoren und Festplatten) und Software (z. B. Betriebssysteme), auf denen eine Unternehmens-IT abläuft. Komplementärprodukte können auch Dienstleis-

---

nicht unerhebliche Kosten (für Schulungen, Software u. Ä.) aufwenden müssen, aber nicht offen legen wollen.

976 Vgl. Kapitel 1.3.1.

977 Vgl. Murphy, Simon /Intangible Benefits/ S. 305; Sassone /CBA of IS/ S. 126.

978 Z. B. kann nur unter großer Unsicherheit festgestellt werden, dass eine Zeitung häufiger gekauft wird, weil eine neue Druckmaschine auch farbig drucken kann und einige Bilder deswegen in Farbe gedruckt werden. Die entsprechenden Kosten – sowohl in der Anschaffung als auch im Unterhalt – können jedoch direkt ermittelt werden und werden der entsprechenden Zeitung zugeordnet.

979 Vgl. Alshawi, Baldwin, Irani /Benchmarking/ S. 416-417; Coombs /Estimating/ S. 2; Coombs beispielsweise stellt fest, dass Unsicherheiten in der Kostenschätzung von IT-Projekten existieren, dass diese allerdings überwunden werden können.

tungen sein, die von einem Produktanbieter als Ergänzung angeboten werden (z. B. Schulungen und Support). Komplementärprodukte werden für jede Unternehmens-IT unabhängig von der eingesetzten Softwarearchitektur benötigt. Jedoch haben sich Hersteller[981] mit einigen Produkten auf die Unterstützung von Unternehmens-IT-Systemen auf Basis einer SOA konzentriert. Mit diesen Produkten ist es den Herstellern möglich, selektive Komplementäreffekte zu erzielen. Selektive Komplementäreffekte erschweren es einem Unternehmen, andere Produkte zur Zusammenarbeit zu verwenden. Dadurch kann es einem Unternehmen erschwert werden, Potentiale entsprechend der Einschätzung dieser Arbeit zu erlangen. Beispielsweise kann eine Integrationsplattform des Herstellers A ggf. keine Services außerhalb einer Unternehmens-IT unterstützen, wenn diese auf einer Plattform des Herstellers B betrieben wird, selbst wenn diese Komponente dafür vorgesehen ist und auf der Plattform des Herstellers A auch extern unterstützt werden könnte. Diese Möglichkeit der Hersteller von Komplementärprodukten kann zu höheren Kosten führen.

Die Infrastruktur einer Unternehmens-IT auf Basis einer SOA umfasst über Komplementärprodukte hinaus die Kommunikationsinfrastruktur.[982] Dazu gehören ein Service-Bus, ein Service-Registry und ein Service-Repository. Eine Kommunikationsinfrastruktur kann ebenso Funktionen enthalten, die z. B. Integrität, Vertraulichkeit, Authentizität, Sicherheit der Kommunikation (z. B. durch Verschlüsselung und Signaturen) und Zugriffe auf Bestandteile einer Unternehmens-IT sicherstellt.[983]

In der Literatur gibt es Berichte, nach denen eine Einführung von EAI-Systemen mit erheblichem Aufwand verbunden ist.[984] Bei einer SOA lässt sich dies durch entstehenden Aufwand zur Einführung und Pflege einer Kommunikationsinfrastruktur erklären. Insbesondere eine schwergewichtige Kommunikationsinfrastruktur, d. h., eine Kommunikationsinfrastruktur, die viele Dienstleistungen und Bestandteile enthält, kann zu hohen Kosten führen.

---

[980] Vgl. zum folgenden Absatz Mellis /Projektmanagement/ S. 27-29.

[981] Beispielsweise TIBCO und IBM.

[982] Vgl. Kapitel 2.2.4.3.

[983] Vgl. Krebs, Thiel /Sicherheit/ S. 152-155; Stahlknecht, Hasenkamp /Wirtschaftsinformatik/ S. 492-494.

[984] Vgl. Aier, Schönherr /Flexibilisierung/ S. 16

Besonders die Kommunikationsinfrastruktur einer SOA als ein Kommunikationsmedium, über das die gesamte Kommunikation einer Unternehmens-IT abgewickelt wird, muss eine hohe Ausfallsicherheit aufweisen und dafür Sorge tragen, dass eine Unternehmens-IT ablauffähig ist. Eine dauerhafte Nicht-Ausführung eines aufgerufenen Dienstes (das Verhungern eines Serviceaufrufs, in der Betriebssystemprogrammierung auch Starvation[985] genannt) darf auch in einer Unternehmens-IT nicht auftreten, weil ansonsten Leistungserstellungsprozesse eines Unternehmens nicht durchgeführt werden können. Auch soll die Kommunikationsinfrastruktur in der Lage sein, ein sehr hohes Datenvolumen abwickeln zu können, was zu Problemen der Zuverlässigkeit und der Performanz führen kann. Aus diesen Gründen müssen nicht nur Investitionen zum Betrieb und Aufbau einer Kommunikationsinfrastruktur, sondern auch entsprechende Investitionen in Tätigkeiten und Ausprägungen einer Kommunikationsinfrastruktur aufgewendet werden, die den aufgezeigten Problemen entgegenwirken.

4.3 Kosten der Entwicklung

Die Softwarekomponenten einer SOA sind Services und Serviceschnittstellen.[986] Diese Komponenten stellen Dienstleistungen einer Unternehmens-IT entsprechenden Dienstleistungsnehmern zur Verfügung. Die in Services und ihren Schnittstellen implementierten Funktionen müssen in jeder Unternehmens-IT – unabhängig von der zugrunde gelegten Softwarearchitektur – implementiert werden. Es entstehen jedoch spezielle Kostenpotentiale durch die Gestaltungsrichtlinien einer SOA. Im Folgenden werden die Auswirkungen der Implementierung von Services (Kapitel 4.3.1) und Serviceschnittstellen (Kapitel 4.3.2) auf die Kosten der Entwicklung untersucht.

4.3.1 Kosten der Entwicklung von Services

Eigenschaften und Gestaltungsziele einer SOA, die Services betreffen, sind: lose Kopplung, Wiederverwendbarkeit, Festlegen des Funktionsumfangs, Verstecken der eigenen Komplexität und Orientierung an Geschäftsprozessen.

---

[985] Mit dem Begriff *starvation* wird das dauerhafte Nichtbearbeiten eines Serviceaufrufs bezeichnet. Vgl. Vogt /Betriebssysteme/ 94.

[986] Vgl. Kapitel 2.2.4.

Services einer SOA sollen lose gekoppelt sein.[987] Eine lose Kopplung muss technisch umgesetzt werden können. Freilich reicht es nicht aus, Technik einzusetzen, die eine lose Kopplung von Softwarekomponenten ermöglicht. Stattdessen müssen Services so gestaltet und implementiert werden, dass eine lose Kopplung möglich ist.[988] Die entsprechenden Aktivitäten in der Analyse, im Design und bei der Implementierung einer Unternehmens-IT verursachen weitere Kosten.[989]

Wiederverwendbarkeit erzeugt höhere Komplexität und Abhängigkeiten der Softwarekomponenten untereinander, welche auch verwaltet werden müssen.[990] Ein universell einsetzbarer Service kann deswegen nicht nur komplexer, sondern auch langsamer ablaufen und schwieriger in der Entwicklung sein. Mehrere potentielle Verwendungsszenarien müssen beachtet, und bei geplanter Wiederverwendung sollen alle Dienstleistungsnutzer zufrieden gestellt werden. Dies kann sogar die erhofften Vorteile der Wiederverwendbarkeit[991] zunichte machen.[992]

Im Zusammenhang mit der Wiederverwendung steht auch das Festlegen des Funktionsumfangs. Gerade der Funktionsumfang kann entscheidend dafür sein, wie oft ein Service wieder verwendet werden kann. Die Festlegung des Funktionsumfangs der Services determiniert allerdings auch, wie viel Kommunikation über die Kommunikationsinfrastruktur zwischen den Services geführt werden muss. Je höher der Funktionsumfang eines Services ist, desto geringer sind der Kommunikationsumfang und die Datenmenge, die mit anderen Services ausgetauscht werden müssen. Der Mehraufwand, um einen angemessenen und wünschenswerten Funktionsumfang festlegen zu können, muss zusätzlich bei der Analyse und dem Design von Services investiert werden.

Das Gestaltungsziel ‚Orientierung an Geschäftsprozessen' erfordert, dass die zu unterstützenden Geschäftsprozesse bekannt sind und analysiert werden. In der Literatur äu-

---

[987] Vgl. Kapitel 2.2.4.3.

[988] Vgl. Sholler /Resuable Enterprise Services/ S. 1-2; Natis, Schulte /Introduction/ S. 2 und S. 5.

[989] Boehm beispielsweise stellt fest, dass die Aktivität zum Erreichen von Portabilität nachteilige Auswirkungen auf die Entwicklungskosten einer Software hat. Vgl. Boehm u. a. /Characteristics/ S. (3-8). Der Aufwand zum Erreichen einer losen Kopplung folgt ähnlichen Argumenten wie der Aufwand zur Erreichung von Portabilität.

[990] Vgl. Sholler /Governance/ S. 2.

[991] Vgl. Kapitel 3.2.2.6.

[992] Sholler berichtet in einer Studie, dass die Wiederverwendungsrate von Services knapp 10% erreicht, wohingegen 15% mehr Entwicklungskosten aufgewendet werden müssen, um wiederverwendbare Softwarekomponenten anbieten zu können. Vgl. Sholler /Case/ S. 2-3.

ßern sich einige Autoren, dass zu diesem Zweck sog. Domänenmodelle[993] der zu unterstützenden Geschäftsprozesse erstellt werden müssen, damit eine Orientierung an Geschäftsprozessen geschehen kann. Auch diese Analyseaktivitäten verursachen Kosten.

4.3.2 Kosten der Entwicklung von Schnittstellen

In einer Unternehmens-IT, in der andere Softwarearchitekturen verwendet werden, werden Schnittstellen häufig als Punkt-zu-Punkt-Verbindung realisiert. [994] Dies führt dazu, dass sich die Anzahl der notwendigen Schnittstellen überproportional mit der Anzahl der Bestandteile einer Unternehmens-IT entwickelt (vgl. Abb. 4-1).[995] Folglich steigen auch die Kosten für Entwicklung und Wartung der Schnittstellen und Anwendungssysteme an.[996] Betreiber einer Unternehmens-IT auf Basis einer SOA äußern oft die Hoffnung, dass die Kosten der Schnittstellenentwicklung und –wartung reduziert werden können, indem einzelne Schnittstellen zur Kommunikationsinfrastruktur, die von allen Servicenehmern verwendet werden, bereit gestellt werden.[997] Dieser Struktur folgend wird eine Serviceschnittstelle von mehreren Dienstleistungsnehmern verwendet, und die Kurve der Kosten der Schnittstellenentwicklung steigt nicht überproportional, sondern linear an und reduziert die Gesamtkosten der Schnittstellen- und Anwendungskosten.

Diese Betrachtung ignoriert jedoch die Notwendigkeit, entsprechende Schnittstellen anbieten zu können. Die Kosten der Erstellung einer allgemeingültigen Schnittstelle sind um ein Vielfaches höher als die Kosten einer spezialisierten Punkt-zu-Punkt-Schnittstelle. Dies begründet sich dadurch, weil Kosten für die Generalisierung einer Schnittstelle entstehen. Zu diesen Kosten gehört zum einen der Aufwand, der entsteht, wenn bei Änderungen und Erstellung einer Serviceschnittstelle entsprechende Abhängigkeiten zu allen (potentiellen) Nutzern beachtet werden müssen.[998] Dadurch steigt der Analyseaufwand einzelner Schnittstellen um ein Vielfaches an. Zum anderen entsteht Aufwand, wenn Schnittstellen angepasst werden müssen. Durch eine Serviceversionisierung ist es möglich, dass bestehende Schnittstellen durch Dienstleistungsnehmer wei-

---

[993] Vgl. Kapitel 2.2.5.2.

[994] Vgl. zu diesem Absatz Chappell /ESB/ S. 80-81.

[995] Vgl. Winter /Modell/ S. 4-5.

[996] Vgl. Winter /Modell/ S. 4-5.

[997] Vgl. Aier, Schönherr /Flexibilisierung/ S. 17; Chappell /ESB/ S. 80-81; Karch u. a. /SAP/ S. 53-54.

terhin verwendet werden können. Jedoch kann sich die Notwendigkeit ergeben, dass Schnittstellen außer Funktion gesetzt werden oder dass durch eine neue Version Vorteile für einen Dienstleistungsnehmer angeboten werden können. Solche Abhängigkeiten können analysiert werden, bedeuten jedoch zusätzlichen Aufwand und können insb. bei vielen Dienstleistungsnehmern zu erheblichen Kosten führen.

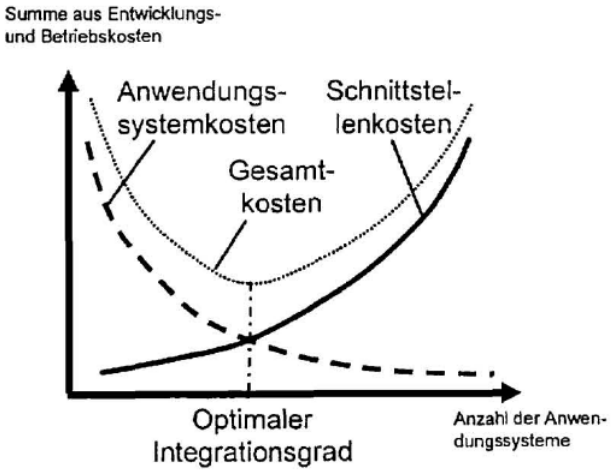

Abb. 4-1: Schnittstellenkosten[999]

Wird eine Unternehmens-IT auf Basis einer SOA implementiert und werden bestehende oder externe Anwendungen über spezielle Schnittstellen (sog. Adapter) eingebunden, entstehen Kosten zur Erstellung der jeweiligen Adapter. Der Aufwand, der zur Erstellung von Adaptern aufgewendet wird, kann unter Umständen erheblich sein, weil das einzubindende Programm analysiert werden muss. Weiterhin muss die Schnittstelle in den Ablauf des Programms eingreifen und den Ablauf eines Programms oder Teile von ihm anstoßen können.[1000]

Ein weiterer Kostenfaktor ist, dass generalisierte Schnittstellen unter Umständen häufiger angepasst werden müssen als es bei Punkt-zu-Punkt-Schnittstellen der Fall ist. Änderungen an generalisierten Schnittstellen werden notwendig, sobald nur ein einziger Dienstleistungsnehmer eine Änderung wünscht. Solche Änderungsanforderungen be-

---

[998] Vgl. Sholler /Governance/ S. 2.

[999] Vgl. Winter /Modell/ S. 4.

[1000] Aier, Schönherr /Flexibilisierung/ S. 16.

trifft bei Punkt-zu-Punkt-Schnittstellen eine einzige, von anderen Schnittstellen und den entsprechenden Dienstleistungsnehmern unabhängige Schnittstelle. Im Fall allgemeiner Schnittstellen müssen Änderungen von allen Nutzern einer Schnittstelle akzeptiert werden. Alternativ können Versionen von Schnittstellen erzeugt werden, und jeder Nutzer einer Schnittstelle kann diese verwenden, die am geeignetsten ist. Beide Alternativen erzeugen jedoch zusätzlichen Aufwand zur Änderung an sich: Im ersten Fall müssen Änderungen allen (potentiellen) Nutzern bekannt gemacht werden und diese ggf. entsprechend angepasst werden. Im zweiten Fall müssen Schnittstellenversionen verwaltet, identifiziert und angesprochen werden können. Betrachtet man die einzelnen Schnittstellen, kann deswegen bei Punkt-zu-Punkt-Schnittstellen auch von beständigeren Schnittstellen gegenüber denen einer SOA gesprochen werden. Wird z. B. eine Änderung einer Schnittstelle bedingt durch einen Nutzer notwendig, so muss sowohl bei allgemeinen als auch Punkt-zu-Punkt-Schnittstellen eine Schnittstelle geändert werden. Bei allgemeinen Schnittstellen ist diese Änderung mit dem aufgezeigten zusätzlichen Aufwand verbunden.

4.4 Kosten durch organisatorischen Wandel

Die Einführung neuer Software bedingt häufig auch organisatorische Änderungen eines Unternehmens.[1001] Zum einen kann dies auf Fachseite, zum anderen auf technischer Seite notwendig werden. Organisatorische Änderungen auf Fachseite – also auf der Seite zu deren Unterstützung die neue Software eingeführt wird, – können z. B. durch neue oder verbesserte Prozesse, die durch die Software möglich sind, bedingt werden. Auf technischer Seite – also auf der Seite, die die Einführung und Wartung der Software durchführt, – können organisatorische Änderungen notwendig werden, weil z. B. neue oder verbesserte Prozesse zur Wartung der Software notwendig bzw. möglich werden.

Ist organisatorischer Wandel nötig, kann dieser auf Verhaltens- und Systemwiderstände treffen.[1002] Widerstände sind dabei auf unterschiedlichen Ebenen (in der Hierarchie ei-

---

[1001] Vgl. Szyperski /Strategisches Informationsmanagement/ S. 8-9; Oxman, Smith /Structual Change/ S. 78.

[1002] Vgl. Macharzina /Unternehmensführung/ S. 495-500; Adam, McKendrick /Everything in Between/ S. 29-31; Macharzina betrachtet in diesem Kontext die Einführung eines strategischen Managements. Die Widerstandstypen lassen sich allerdings problemlos auf die Einführung neuer allgemeiner Organisationsstrukturen übertragen, weil die Einführung eines strategischen Managements einen Spezialfall des organisatorischen Wandels darstellt.

nes Unternehmens), durch unterschiedliche Stellen (z. B. des betroffenen Fachbereichs oder der IT-Abteilung) und aus weiteren Gründen (z. B. Änderung des Aufgaben-, Macht- oder Einflussbereichs) möglich.[1003] Um Widerstände zu überwinden sind Aktivitäten, z. B. Coaching[1004] und Drittparteien-Interventionen, notwendig.[1005] Außerdem sind entsprechende Strukturen und organisatorische Instanzen zur Institutionalisierung des Wandels zu bilden.[1006] Die Aktivitäten, um Widerstände zu überwinden, und die Institutionalisierung von Wandel verursachen Kosten.

Im Folgenden wird auf organisatorische Strukturen eingegangen, die aufgrund des Einsatzes einer SOA einem Wandel unterworfen sind bzw. sein können und dadurch Kosten verursachen (können). Unterschieden wird dabei zwischen Wandel der Aufbauorganisation (Kapitel 4.4.1) und der Ablauforganisation (Kapitel 4.4.2).[1007]

4.4.1 Kosten durch Wandel der Aufbauorganisation

Die organisatorischen Strukturen können durch technologische Veränderungen nicht unberührt bleiben.[1008] Änderungen dieser Strukturen betreffen dem Bedienungs- und Implementationsrahmen der Unternehmens-IT. Veränderungen können sowohl die Strukturierung von Abteilungen als auch die Einführung neuer Stellen betreffen.

Rollen, welche bei der Implementierung einer SOA eingeführt werden sollten, sind:[1009]

- Unternehmensweite Softwarearchitekten

  Unternehmensweite Softwarearchitekten definieren die unternehmensweite Soft-

---

[1003] Vgl. zur Thematik von Widerständen in der Informationstechnik beispielsweise Raasch /Systementwicklung/ S. 12-13; Mellis /Projektmanagement/ S. 254-255.

[1004] Adam und McKendrick beispielsweise stellen fest, dass die Ausbildung der Software-Entwickler zur Entwicklung einer SOA heute noch nicht ausreichend ist. Vgl. Adam, McKendrick /Everything in Between/ S. 29-31.

[1005] Zu einer Übersicht über Strategien und Aktivitäten, um Widerstand begegnen zu können vgl. Mellis /Projektmanagement/ S. 241-253.

[1006] Vgl. Aier, Schönherr /Flexibilisierung/ S. 9.

[1007] Die Organisationsgestaltung kann in Aufbau- und Ablauforganisation unterschieden werden, vgl. Kapitel 3.2.3.2.1. Vgl. Macharzina /Unternehmensführung/ S. 352; Mellis /Projektmanagement/ S. 80.

[1008] Vgl. Szyperski /Strategisches Informationsmanagement/ S. 8-9.

[1009] Vgl. Huizen /SOA/ S. 27. Nach Huizen unterliegen Projekte mit kleinem Umfang ggf. nicht der Anforderung, alle diese Rollen einzuführen, und es kann durchaus der Fall sein, dass Rollen auf ein Team bzw. einzelne Mitarbeiter zusammengefasst werden.

warearchitektur, wählen Technologien, arbeiten Standards und Richtlinien aus und unterrichten Entwickler und andere Beteiligte von ihren Arbeitsergebnissen.

- Architekten zur Orchestration von Services

  Diese Architekten untersuchen Anforderungen und Geschäftsprozesse, definieren Services und stellen diese zur Verwendung in der Unternehmens-IT entsprechend zusammen, damit diese die Anforderungen erfüllen können.

- Bibliothekar für Services

  Ein Bibliothekar verwaltet alle Services (bzw. Softwarekomponenten) einer Unternehmens-IT. Er stellt sicher, dass Services entsprechend dokumentiert und Informationen richtig veröffentlicht sind. Ferner soll sichergestellt werden, dass Versionen von Services korrekt erstellt werden, und dass sie mit der bestehenden Unternehmens-IT ohne Probleme zusammenarbeiten können.

- Integrationsteam

  Das Integrationsteam ist dafür verantwortlich, dass die neu erstellte Software und organisatorische Prozesse gleichzeitig mit bisher benutzter Software, ihren Daten sowie den bisher verwendeten Prozessen eingesetzt werden. Erforderlichenfalls muss durch das Team sichergestellt werden, dass auch eine Zusammenarbeit neuer und alter Software möglich ist. Als hauptsächliche Integrationsgrundlage ist das Integrationsteam auch für den Betrieb der Kommunikationsinfrastruktur verantwortlich.

- Serviceentwickler

  Die Software-Entwickler sind für die Erstellung vom Funktionen zur Anforderungserfüllung verantwortlich. Sie erstellen Services einer Unternehmens-IT und Adapter zur Anbindung von Programmen an die Unternehmens-IT.

In jedem Unternehmen, das Software zur Unterstützung der Dienstleistungserstellung einsetzt, sollten die Rollen der unternehmensweiten Softwarearchitekten, eines Integrationsteams und der Software-Entwickler vorhanden sein und auch entsprechend ausgefüllt werden.[1010] Die Rollen der Architekten zur Orchestration und des Bibliothekars für Services jedoch sind nicht in entsprechenden Ausprägungen zu erwarten und werden somit als spezifische Rollen einer SOA eingestuft. Diese Rollen einzuführen und ent-

---

[1010] Mellis beschreibt, dass Rollen zur Softwareentwicklung vorhanden sein sollen, vgl. Mellis /Projektmanagement/ S. 12. Rollen, die je nach Größe eines Projektes umgesetzt werden sollen, werden z. B. beschrieben in Zuser, Grechenig, Köhle /Software Engineering/ S. 170-187.

sprechendes Personal für diese Rollen abzustellen verursacht sowohl wiederkehrende (Personalkosten) als auch nicht-wiederkehrende Kosten (Einführungskosten).

4.4.2 Kosten durch Wandel der Ablauforganisation

Untersucht man den Wandel der Ablauforganisation müssen organisatorische Anforderungen der durch eine Unternehmens-IT zu unterstützenden Geschäftsprozesse[1011] sowie der Prozesse der Entwicklung und des Betriebs einer SOA beachtet werden.

Wandel der Ablauforganisation, der durch die unterstützten Geschäftsprozesse bedingt ist, betrifft die Fachabteilungen eines Unternehmens. Ein Wandel solcher Geschäftsprozesse findet ggf. bei jeder Änderung oder Neueinführung von Software statt und ist nicht von einer spezifischen Softwarearchitektur abhängig. Infolgedessen entstehen keine direkten Kosten durch Änderungen der Ablauforganisation auf Seite der Fachabteilungen, die durch die Verwendung einer SOA bedingt sind.

Wandel der Ablauforganisation, der durch die Prozesse der Entwicklung und des Betriebs einer SOA bedingt ist, betrifft die IT-Abteilung und verursacht Änderungen im Entwicklungsprozess einer Unternehmens-IT. Dies beinhaltet z. B. die Koordination der verschiedenen Abteilungen eines Unternehmens, die zusammenarbeiten und sich dadurch bezüglich der zu unterstützenden Dienstleistungen abstimmen müssen. Weiterhin muss die Wiederverwendung einzelner Services erkannt und durchgesetzt werden.[1012] Hierunter fällt auch die Verwendung und Pflege der Kommunikationsinfrastruktur, die auf die Prozesse der IT-Abteilung Einfluss hat. Solche Anforderungen haben sowohl nicht-wiederkehrende als auch wiederkehrende Kosten zur Folge.

4.5 Kosten der Ablösung und Einbindung von Legacy-Systemen

Bei der Einführung einer SOA zur Ablösung von bestehenden Alt-Systemen, sog. Legacy-Systemen,[1013] entsteht ein hoher Aufwand, weil eine Kommunikationsinfrastruktur aufgebaut werden muss, die als Basis für die zu erstellende Unternehmens-IT dienen kann und zukünftigen Entwicklungen gewachsen ist. Ein zusätzlicher Aufwand entsteht

---

[1011] Vgl. Kaib /EAI/ S. 97; Picot, Reichwald, Wiegand /Grenzenlose Unternehmung/ S. 195-196.

[1012] Vgl. zu diesem Absatz Sholler /Resuable Enterprise Services/ S. 2; Sholler /Governance/ S. 1.

[1013] Vgl. Sommerville /Software Engineering/ S. 589-606; Inmon, Zachman, Geiger /Data/ S. 33.

durch eine Analyse der bestehenden Geschäftsprozesse zur Erstellung eines Domänenmodells, welches der Bestimmung des Funktionsumfangs einzelner Services dient.[1014] Solche Ausgaben können nur teilweise spezifisch einer SOA zugerechnet werden, weil bei einer Ablösung von Legacy-Systemen die Chance wahrgenommen wird, neben der Unternehmens-IT auch organisatorische Prozesse anpassen zu können.[1015] Eventuell notwendige Änderungen an Legacy-Systemen, die besonders kostspielig sind,[1016] können vergleichbare Kosten verursachen.

Müssen oder sollen Legacy-Systeme eingebunden werden, ist es i. d. R. notwendig, geeignete Adapter zu entwerfen und entwickeln. Diese müssen nach außen das Verhalten von Services darstellen (z. B. die entsprechenden Schnittstellen anbieten und mit der Kommunikationsinfrastruktur kommunizieren[1017]). Die Dienstleistung wird aber durch die existierende Anwendung erbracht. Durch ein solches Vorgehen können existierende Dienstleistungen weiterhin verwendet werden. Die entsprechenden Kosten sind jedoch nicht spezifisch einer SOA zurechenbar. Solche Kosten würden immer auftreten, wenn entsprechende Legacy-Systeme weiterhin verwendet werden sollen bzw. müssen – unabhängig von der verwendeten Softwarearchitektur.

4.6 Kosten des Managements einer SOA

Wird die Unternehmens-IT neu implementiert bzw. gewartet und wird hierzu ein neues Entwicklungskonzept eingesetzt, können daraus neue Anforderungen an das Management der Softwareentwicklung resultieren.[1018] Wie in Kapitel 4.4 aufgezeigt, gehört hierzu die Einführung organisatorischer Rollen wie z. B. die eines Service-Bibliothekars.[1019] Aufgaben des Managements können in zwei Bereiche eingeteilt werden: In technisch-fachliche und in administrative Aufgabenbereiche.

[1014] Vgl. Kapitel 2.2.5.2.

[1015] Vgl. Krüger, Seelmann-Eggbert /IT-Architektur/ S. 211-217.

[1016] Vgl. Sommerville /Sofware Engineering/ S. 590-591.

[1017] Vgl. für weitere Eigenschaften Kapitel 2.2.4.

[1018] Vgl. Mellis /Projektmanagement/ S. 241.

[1019] Vgl. zu den verschiedenen Rollen Huizen /SOA/ S. 27. Huizen sagt, dass bei kleineren SOA-Projekten ggf. nicht alle Rollen erforderlich sind oder durch eine oder wenige Personen ausgeführt werden können, dass eine Einführung aller Rollen jedoch eine erfolgreiche Einführung einer SOA versichert.

Technisch-fachliche Aufgabenbereiche betreffen das Management der Unternehmens-IT, im Falle einer SOA somit die Services, Schnittstellen und die Kommunikationsinfrastruktur.[1020] Diese Bestandteile der Struktur einer SOA haben jeweils einen eigenen Lebenszyklus, was wiederum bedeutet, dass jeder Bestandteil unabhängig von anderen Bestandteilen einer Unternehmens-IT gemanagt werden muss.[1021] Das Management der Bestandteile einer Unternehmens-IT wird Konfigurationsmanagement genannt.[1022]

Das Konfigurationsmanagement einer SOA erzeugt Kosten, die spezifisch einer SOA zugeordnet werden können, weil jeder Service, jede Schnittstelle und die Kommunikationsinfrastruktur (wiederum mit ihren Bestandteilen[1023]) zusammen betrachtet werden müssen. Teile dieser Aufgaben können durch Zuhilfenahme der vorhandenen Struktur unterstützt werden und führen zu einer Verringerung der spezifischen Kosten. In einer Unternehmens-IT auf Basis einer SOA müssen z. B. die Existenz und Verfügbarkeit einzelner Services verwaltet werden.[1024] Ein Service-Repository kann genau diese Aufgabe übernehmen[1025] und kann zudem, unterstützt durch ein Service-Registry, semantische Anfragen auswerten und beantworten.[1026]

Zum Konfigurationsmanagement gehört die Verwaltung von ‚Service Contracts' (Serviceverträgen). Die Definition der Serviceschnittstellen zusammen mit beschreibenden Metadaten[1027] der angebotenen Dienstleistung eines Services wird ‚Service Contract' genannt.[1028] Ein solcher Service Contract muss öffentlich bekannt sein, damit Dienstleistungsnachfrager die angebotene Dienstleistung beurteilen und auffinden können.[1029]

[1020] Vgl. Kapitel 2.2.4.

[1021] Vgl. Sholler /Resuable Enterprise Services/ S. 1.

[1022] Vgl. Sommerville /Software Engineering/ S. 651-653; Zuser, Grechenig, Köhle /Software Engineering/ S. 326; Hansen /Wirtschaftsinformatik/ S. 152-153 (Hansen stellt das Konfigurationsmanagement des V-Modell 97 dar.)

[1023] Vgl. Kapitel 2.2.4.3.

[1024] Vgl. Adam, McKendrick /Everything in Between/ S. 3, 13-14.

[1025] Vgl. Adam, McKendrick /Everything in Between/ S. 3 und S. 13-14.

[1026] Vgl. Kapitel 2.2.4.3.

[1027] Beschreibende Metadaten eines Services umfassen beispielsweise: Performanz, Verarbeitungskapazität, verantwortliche Stelle, bekannte Fehler, Verlässlichkeit, (vorausgesetzte) Sicherheitsebene und -funktionen, Vor- und Nachbedingungen sowie ggf. Abhängigkeiten (z. B. von Daten oder externen Funktionen) Vgl. Banke, Krafzig, Slama /Enterprise SOA/ S. 63.

[1028] Vgl. Banke, Krafzig, Slama /Enterprise SOA/ S. 59; Sholler /Resuable Enterprise Services/ S. 2; Chappell /ESB/ S. 53-55.

[1029] Vgl. Banke, Krafzig, Slama /Enterprise SOA/ S. 59-60.

Z. B. können Serviceentwickler beurteilen, ob eine angebotene Dienstleistung den eigenen Anforderungen genügt.[1030] Die Aufgabe, einen Service Contract zu verwalten und die Informationen bereitzustellen, wird von einem Service Bibliothekar beaufsichtigt. Dieser wird hierbei auch durch die Kommunikationsinfrastruktur unterstützt.

Zum Konfigurationsmanagement einer SOA gehört auch die Orchestrierung verschiedener Services. Im Unternehmensumfeld wird dies auch als Business Process Management (BPM) bezeichnet. Das BPM einer Unternehmens-IT auf Basis einer SOA ist für den Aufruf mehrerer Services zuständig, die in ihrer Gesamtheit und Reihenfolge Geschäftsprozesse abbilden. Weil eine SOA die Orchestrierung auch zu späteren Zeitpunkten als dem Erst-Design einer Unternehmens-IT zulässt, kann diese Aufgabe auch zu jedem späterem Zeitpunkt geschehen und ist mit zusätzlichen Kosten verbunden.

Administrative Aufgaben betreffen das Management des Entwicklungsprozesses. Dies sind vor allem Querschnittsaktivitäten[1031] wie Leitung, Kontrolle und Qualitätssicherung.[1032] Es kann davon ausgegangen werden, dass sich administrative Aufgaben einer SOA nicht wesentlich von denen unterscheiden, wenn eine Unternehmens-IT auf Basis einer anderen Softwarearchitektur eingesetzt wird. Die Aufgaben der Leitung und Kontrolle sowie der Qualitätssicherung sind bei jeder Entwicklung und Wartung einer Unternehmens-IT vorhanden.[1033] Die Kosten der administrativen Aufgaben sind abhängig von dem Umfang entsprechender Projekte, aber nicht von der Art der eingesetzten Softwarearchitektur.

[1030] Vgl. Sholler /Governance/ S. 1.

[1031] Entwicklungsaufgaben sind unmittelbar auf die Erreichung des Sachziels ausgerichtet. Ihr Fokus ist die Entwicklung eines Softwareprodukts. Unterstützungsaufgaben dagegen helfen den Aufgabenträgern bei der Bewältigung von Entwicklungsaufgaben. Vgl. Stelzer /Möglichkeiten/ S. 99.

[1032] Vgl. Seibt /Vorgehensmodell/.

[1033] Vgl. Zuser, Grechenig, Köhle /Software Engineering/ S. 76-77; Sommerville /Software Engineering/ S. 545-546;

5. Wirtschaftlichkeitsanalyse einer SOA

Die Einführung einer SOA wird in dieser Arbeit als Ansatz verstanden, um zu einer ganzheitlich integrierten Unternehmens-IT zu gelangen.[1034] Eine SOA stellt folglich eine Gestaltungsalternative einer Enterprise Application Integration (EAI) dar. Die EAI muss im Gesamtunternehmenskontext betrachtet werden. Es wird angenommen, dass sie bei einer solchen holistischen Betrachtungsweise einen der wichtigsten Erfolgsfaktoren für eine integrierte Unternehmens-IT darstellt.[1035] In diesem Kapitel wird eine holistische Betrachtung zur Bewertung einer SOA angestrebt. Diese geht von Faktoren aus, die die Wirtschaftlichkeit eines Unternehmens beeinflussen, und sie stellt die herausgearbeiteten Nutzen- und Kostenpotentiale mit diesen Faktoren in Beziehung. Dabei werden einzelne Teilbereiche identifiziert, die im Zusammenhang mit einer Softwarearchitektur bewertet werden können. Zu diesen Teilbereichen werden qualitative Bewertungen auf Basis einer Argumentenkette bzw. –sammlung anhand einer Ordinalskala vorgenommen.

5.1 Problematik der Bewertung der Wirtschaftlichkeit

Die Problematik der Bewertung der Wirtschaftlichkeit einer SOA wird anhand folgender Argumente verdeutlicht:

- Bewertungen zur Entscheidungsgrundlage müssen situationsbezogen sein,
- Unsicherheit der Bewertung aufgrund eines langfristigen Zeithorizonts,
- Unsicherheit der Bewertung aufgrund ungenauer Bewertbarkeit von Qualitätsattributen,
- Unsicherheit der Bewertung aufgrund mehrerer Bewertungsdimensionen,
- Unsicherheit der Bewertung aufgrund unterschiedlicher Interpretation der Gestaltungsziele und Bewertungsdimensionen sowie
- Unsicherheit der Bewertung aufgrund der Eigenschaft einer ex-ante Wirtschaftlichkeitsbewertung, keine genauen Vorhersagen treffen zu können.

[1034] Vgl. Kapitel 2.3.1

[1035] Aier, Schönherr /Flexibilisierung/ S. 5.

Die Ausprägung der Wirtschaftlichkeit einer Unternehmens-IT kann nur in Bezug zur Situation der entsprechenden Unternehmens-IT bewertet werden. Der Grund dafür ist, dass die Unternehmen verschiedene Anforderungen an ihre Unternehmens-IT stellen. Eine einheitliche Bewertung einer Unternehmens-IT für alle Unternehmen ist deswegen nicht möglich. Die jeweiligen Anforderungen leiten sich z. B. von den jeweiligen strategischen Zielen sowie von der spezifischen Markt- und Wettbewerbssituation eines Unternehmens ab.[1036] Weil eine Unternehmens-IT nach Gestaltungsrichtlinien einer Softwarearchitektur aufgebaut ist, gelten die entsprechenden unterschiedlichen Anforderungen an eine Unternehmens-IT auch für eine Softwarearchitektur. Deswegen kann eine einheitliche Bewertung einer SOA, die für alle Unternehmen gleichermaßen gültig ist, nicht aufgestellt werden. Beispiele für Anforderungen, deren Wichtigkeit in verschiedenen Unternehmen unterschiedlich ausgeprägt sein kann, sind Projektaufwand und Qualitätsanforderungen.[1037] Demgegenüber existieren Anforderungen, die für alle Unternehmen in Bezug auf die jeweilige Unternehmens-IT eine ähnliche Wichtigkeit haben. Beispiele hierfür sind Wiederverwendbarkeit[1038] und Infrastrukturkosten sowie -komplexität[1039].

Die Entscheidung für den Einsatz einer Softwarearchitektur im Unternehmen stellt eine Entscheidung mit langfristigem Zeithorizont dar. Zum einen wird die Unternehmens-IT in Unternehmen über viele Jahre eingesetzt;[1040] zum anderen kann ein Unternehmen eine existierende Unternehmens-IT nicht kurzfristig durch eine andere ersetzen.[1041] Die Unternehmens-IT muss i. d. R. lange betrieben und gewartet werden. Eine langfristige

---

[1036] Vgl. Karch u. a. /SAP/ S. 55-56.

[1037] Vgl. z. B. Zahavi /Integration/ S. XLIV-XLV; Kaib /EAI/ S. 22; Themistocleous, Irani /Benchmarking/ S. 325-328. Der Zusammenhang kann anhand des sog. ‚Spannungsdreiecks' (vgl. Posch, Birken, Gerdom /Basiswissen/ S. 5) verdeutlicht werden: Die Parameter Termine, Budget und Qualität stehen in Beziehung zueinander. Durch die Festlegung der Wichtigkeit einzelner Parameter wird ein grundlegendes Vorgehen angestrebt: z. B. kann der Termin unabänderlich sein, wodurch Einfluss auf Kosten und Qualität eines Produktes ausgeübt werden kann. Dem ‚Spannungsdreieck' zählen einige Autoren noch den Funktionsumfang ergänzend hinzu. Vgl. beispielsweise Mellis /Projektmanagement/ S. 15-19 (Mellis redet zwar nicht von einem ‚Spannungsdreieck', bringt die vier Dimensionen allerdings in den gleichen Kontext).

[1038] Vgl. Linthicum /Application Integration/ S. 61; Zahavi /Integration/ S. 78-79; Kaib /EAI/ S. 22-34; Cummins /Enterprise Integration/ S. 23-44, 423-424; Ruh, Maginnis, Brown /EAI/ S. 3-7, S. 12, S. 20 & S. 156; Themistocleous, Irani /Benchmarking/ S. 325-328.

[1039] Vgl. Kaib /EAI/ S. 22-34; Cummins /Enterprise Integration/ S. 423-424; Themistocleous, Irani /Benchmarking/ S. 325-328.

[1040] Vgl. Sommerville /Sofware Engineering/ S. 589; Kirchmer /Einführung/ S. 43.

[1041] Vgl. Gruhn, Thiel /Komponentenmodelle/ S. 15.

Bewertung der Auswirkungen auf die Softwareentwicklung wird dadurch notwendig. Weil eine Unternehmens-IT über lange Zeiträume betrachtet kontinuierlich weiterentwickelt wird und neue Anforderungen an die Unternehmens-IT gestellt werden,[1042] bergen solche Bewertungen Unsicherheiten. Deshalb kann von einer Bewertungssituation unter Unsicherheit ausgegangen werden.

Die Unsicherheit einer Bewertung kann auch wegen ungenauer Bewertbarkeit der Qualitätsattribute, die zur Bewertung herangezogen werden, eintreten. Wie in den Kapiteln 3.2.1 und 4.1 schon dargestellt, können sowohl Nutzen- als auch Kostenpotentiale auch intangible Eigenschaften aufweisen und deswegen nicht eindeutig bestimmt werden.[1043]

Bei der Bewertung der Wirtschaftlichkeit sollten nicht nur Kostenaspekte[1044] betrachtet werden, sondern die identifizierten Kostenpotentiale sollen den operativen und strategischen Nutzenpotentialen gegenübergestellt werden.[1045] Zum einen erhöht eine solche Betrachtung die Komplexität der Bewertung, weil mehrere Dimensionen bewertet und gegenübergestellt werden müssen; zum anderen ist eine genaue Bewertung der Wirtschaftlichkeit nur für eine spezifische, zugrunde gelegte Situation ausführbar.[1046] Die Bewertung und Analyse der Kosten- als auch Nutzenpotentiale ist demzufolge nicht allgemeingültig durchführbar. In dieser Arbeit werden Kriterien erarbeitet, die für Unternehmen relevant sind, von denen jedoch nicht alle für jedes Unternehmen relevant sein müssen. Die Wirtschaftlichkeit einer SOA wird hier unter der Annahme bewertet, dass alle aufgestellten Kriterien für eine Bewertungssituation relevant sind. In ihrer Gesamtheit stellen sie die hier zugrunde gelegte Situation zur Bewertung dar und müssen in der Praxis entsprechend der jeweiligen Situation ausgewählt werden.

Eine Softwarearchitektur anhand einer unterstellten Bewertungssituation zu bewerten, ist nur dann sinnvoll, wenn diese Bewertungssituation entweder für alle oder zumindest

---

[1042] Vgl. Mellis /Softwaremanagement/ S. 171. Mellis bezieht sich in seinen Ausführungen auf Standardsoftware. Eine Übertragung dieser Aussage auf die Entwicklung einer Unternehmens-IT kann jedoch vorgenommen werden, weil die Grundaussage, dass Software über Jahre betrachtet weiterentwickelt wird, sowohl für Standardsoftware als auch für eine Unternehmens-IT zutrifft. Insbesondere für den Fall der Betrachtung von Standardsoftware für Unternehmen ist diese Argumentation zutreffend.

[1043] Heimann und Kappes stellen dies explizit für eine Kostenbewertung dar. Vgl. Heimann, Kappes /Kostenfalle/ 47-49.

[1044] Hierunter fallen sowohl Einsparmöglichkeiten als auch höhere Kosten.

[1045] Karch u. a. /SAP/ S. 25.

[1046] Vgl. Kapitel 1.2.

für viele Unternehmen Gültigkeit besitzt; oder wenn sie derart in Bestandteile aufgeteilt werden kann, dass eine Bewertung durch Auswahl relevanter Bewertungskriterien für alle Unternehmen möglich ist. Folgt man der letztgenannten Bewertungssituation, müssten spezifische Anforderungen von Unternehmen beachtet werden, die jedoch nur für einen Bruchteil von Unternehmen relevant sind. Dies jedoch würde den Rahmen dieser Arbeit um ein Vielfaches übersteigen.[1047] Folglich kann nur eine Bewertungssituation zugrunde gelegt werden, die für möglichst viele Unternehmen Gültigkeit besitzt. Eine solche Bewertung kann nur Tendenzen darstellen, weil nicht alle Kriterien ausgewertet werden können. Eine solche tendenzielle Aussage kann getroffen werden, weil Unternehmen oft ähnlichen Anforderungen zugrunde liegen.[1048]

Der in dieser Arbeit verfolgte Ansatz, um die Wirtschaftlichkeit einer SOA auf Basis einer allgemein gültigen Bewertungssituation zu bewerten, beruht auf der Möglichkeit, jedes identifizierbare Wirtschaftlichkeitspotential einer Person oder Gruppe zuordnen zu können.[1049] Diese Personen oder Gruppen sind Kunden, Mitarbeiter, Konkurrenten, Geschäftspartner und das Unternehmen.[1050] Ein alternativer Ansatz kann nach Iansiti und Levien identifiziert werden:[1051] Um eine weitere Bewertung vorzunehmen, muss untersucht werden, welche spezifischen Wirkungen gegenüber potentiellen Systemen[1052], zu denen eine Unternehmens-IT interoperabel sein soll, existieren.[1053] Iansiti und Levien stellen einerseits fest, dass es nicht möglich ist, eine präzise Grenze zwischen einem Unternehmen und dem Ökosystem des Unternehmens zu ziehen; andererseits sind Ökosysteme von Unternehmen zu Unternehmen verschieden ausgeprägt.[1054] Daher kann die Bewertung nur auf Gruppen von Beteiligten oder auf Teilsysteme eines potentiellen

---

[1047] Für Beispiele zur Bewertung bzw. Analyse einer SOA für spezifische Unternehmen bzw. Industriezweige vgl. z. B. Bath, Herr /Post/ (Deutsche Post World Net), Schlamann /Service Orientation/ und Hagen /Credit Suisse/ (beide Credit Suisse).

[1048] Vgl. Laudon, Laudon /Business/ S. 35-40.

[1049] Vgl. Thorp /Information Paradox/ S. 42-53. In diesem Sinne können auch Unternehmen als Personen bzw. Gruppen angesehen werden.

[1050] Stakeholder eines Unternehmens sind z. B. Angestellte, Besitzer, Kunden, Lieferanten. Vgl. Wolfe, Putler /Stakeholder Groups/ S. 65; Macharzina /Unternehmensführung/ S. 9.

[1051] Vgl. zum folgenden Absatz Iansiti, Levien /Strategy/ S. 71.

[1052] (Anm. des Autors:) An dieser Stelle sind keine IT-Systeme, sondern vielmehr Bestandteile des Ökosystems eines Unternehmens gemeint.

[1053] Macharzina nennt die diese Aufgabe die „Bewältigung der Umweltanforderungen durch das Unternehmen“. Vgl. Macharzina /Unternehmensführung/ S. 8. Zum Ökosystem eines Unternehmens vgl. Macharzina /Unternehmensführung/ S. 14-27; Jurkovich /Organisational Environment/ S. 380-394.

Ökosystems zurückgreifen. Nach Iansiti und Levien können diese als Lieferanten, Kunden, Tochter- und Outsourcing-Unternehmen, Ämter und Wettbewerber identifiziert werden.[1055] Laudon und Laudon kommen zu einem sehr ähnlichen Ergebnis und identifizieren Lieferanten, Kunden, Teilhaber, Regulatoren und Wettbewerber.[1056] Lieferanten und Outsourcing-Unternehmen sind Geschäftspartner eines Unternehmens. Tochterunternehmen sind Bestandteil des Unternehmens. Ämter können ebenso als Geschäftspartner aufgefasst werden, weil Beziehungen zu Ämtern vergleichbar mit Beziehungen zu Geschäftspartnern sind: Ämter stellen Anforderungen, die das Unternehmen erfüllen muss, will es mit dem Amt weiterhin zusammenarbeiten. Der einzige Unterschied ist, dass i. d. R. eine gesetzlich vorgeschriebene Zusammenarbeitspflicht gegenüber Ämtern (z. B. dem Steueramt) besteht.

Die Bewertung der Wirtschaftlichkeit orientiert sich entsprechend dieser Argumentation an den folgenden Kriterien:

- Kundenzufriedenheit[1057] (Kunden)
- Mitarbeiterzufriedenheit (Mitarbeiter)
- Wettbewerbssituation (Konkurrenz)
- Beziehung zu Geschäftspartnern (Geschäftspartner) und
- Interne Wirkungen (Unternehmen)

Dieses Vorgehen allerdings wirft wiederum das Problem der Interpretationsabhängigkeit der Bewertung auf. Verschiedene Personen bzw. Gruppen können differierende Meinungen auf die Wichtigkeit einzelner Qualitätsattribute haben, wodurch Bewertungen zu ungleichen Ergebnissen führen können.[1058] Andererseits können verschiedene Ausprägungen des Wissens zu Bewertungsvorgehen und des Verständnisses über das zu bewertende Objekt vorliegen. Dies kann dazu führen, dass Bewertungen einzelner Per-

---

[1054] Vgl. zum folgenden Absatz Iansiti, Levien /Strategy/ S. 71.

[1055] Vgl. Iansiti, Levien /Strategy/ S. 68 & S. 70-71.

[1056] Vgl. Laudon, Laudon /Business/ S. 38-40.

[1057] Mellis beispielsweise argumentiert die Wichtigkeit der Kundenzufriedenheit für einen Softwarehersteller. Vgl. Mellis /Projekmanagement/ S. 18-19. Die Argumentation von Mellis, dass eine hohe Kundenzufriedenheit wichtig zur Akquise von Folgeaufträgen ist, lässt sich zumindest auf Dienstleistungsunternehmen und Unternehmen der Konsumgüterindustrie übertragen, da diesen vergleichbaren Mechanismen für Dienstleistungen und Produkte zugrunde liegen.

[1058] Vgl. zum folgenden Absatz Johansson /Quality Requirenments/ S. 9066.

sonen oder Gruppen unterschiedlich ausfallen. Der Nutzen einer bestimmten Ausprägung der Unternehmens-IT ist z. B. für einen Kunden wesentlich anders als für das Unternehmen. Deshalb wird die Wirtschaftlichkeit immer aus der Sicht des Unternehmens durchgeführt.[1059]

Die Problematik der Interpretationsabhängigkeit kann durch die vorliegende Arbeit jedoch nicht vollständig ausgeschlossen werden, weil Unternehmen bzw. die Bewerter eines Unternehmens die Beurteilung der Wichtigkeit und die Ausprägung einzelner Qualitätsattribute unterschiedlich festlegen können. Unter den getroffenen Annahmen wird dennoch ein für alle Unternehmen gültiges Gesamtergebnis erreicht. Jedenfalls werden sowohl das (a) Bewertungsvorgehen als auch das (b) Bewertungsobjekt in dieser Arbeit einheitlich gehandhabt. Das Bewertungsvorgehen beruht in der gesamten Arbeit auf dem kritischen Realismus. Zum Bewertungsobjekt wurde in Kapitel 2 eine Arbeitsdefinition aufgestellt.

5.2 Wirtschaftlichkeitspotential

Wie in der Einleitung zu Kapitel 1 dargestellt wird das Wirtschaftlichkeitspotential einer SOA in den folgenden Kapiteln 5.2.1 bis 5.2.5 anhand des Nutzen- und des (operativen und strategischen) Kostenpotentials betriebswirtschaftlich relevanter Wirtschaftlichkeitskriterien beurteilt. Potentieller Nutzen kann sich dabei aus quantitativen und qualitativen Nutzenpotentialen ergeben.[1060] Quantitative Nutzenpotentiale sind z. B. Umsatzsteigerung, Erhöhung des Marktanteils, Kostensenkung, Gewinnerhöhung, Verbesserung der Gesamt-, Kapital- oder Arbeitsproduktivität, der Investitions-, Kapital- oder Umsatzrentabilität.[1061] Zu den qualitativen Nutzenpotentialen gehören: Erhöhung der Differenzierung gegenüber Konkurrenten, Erhöhung der Kundenbindung, Erschließung neuer Märkte bzw. Geschäftsfelder, Verbesserung der Qualität der Kundenberatung, Möglichkeit der schnelleren Reaktion auf Änderungen des Marktes und die Erhöhung der Marktrelevanz gegenüber Lieferanten.

[1059] Die Bewertung wird auch *nur* aus der Sicht des Unternehmens durchgeführt, da die Wirtschaftlichkeit aus Unternehmenssicht bewertet wird. Vgl. Kapitel 1.

[1060] Vgl. zum folgenden Absatz Macharzina /Unternehmensführung/ S. 649-651.

[1061] Vgl. auch Potthof /Empirische Studien/ S. 55-56.

Die Bewertungen in den folgenden Kapiteln beleuchten im Speziellen Auswirkungen einer SOA. Effekte, die eine neutrale Auswirkung auf die Wirtschaftlichkeit haben und deswegen eine Bewertung nicht beeinflussen würden, werden nicht explizit aufgeführt. Für die Bewertung anderer Softwarearchitekturen oder für eine spezifische Bewertung in der Praxis können diese neutralen Effekte jedoch durchaus in eine positive oder negative Richtung ausschlagen.[1062]

5.2.1 Kundenzufriedenheit

In Kapitel 5.1 wurde aufgezeigt, dass Wirtschaftlichkeitspotential unter anderem anhand der Kundenzufriedenheit argumentiert werden kann. Im folgenden Kapitel 5.2.1.1 wird zunächst der Grund verdeutlicht, warum durch den Einsatz einer SOA die Kundenzufriedenheit beeinflusst wird. Anschließend wird in Kapitel 5.2.1.2 dargestellt, welche Auswirkungen eine SOA auf die Kundenzufriedenheit aufweist.

5.2.1.1 Einfluss einer SOA auf Kundenzufriedenheit

Mehrere Autoren sehen als Hauptziel von Aktivitäten zur Integration aller Bestandteile einer Unternehmens-IT die Anpassung an Erfordernisse des Marktes.[1063] Eine SOA ermöglicht den verteilten Einsatz einer Unternehmens-IT.[1064] Dadurch lässt sich eine EAI mittels einer SOA unternehmensweit und über Unternehmensgrenzen hinweg durchführen. Diese Fähigkeit der Unternehmens-IT ermöglicht eine Fokussierung auf Erfordernisse des Marktes, z. B. um neue Faktoren- oder Konsumentenmärkte erschließen zu können. Auch die Orientierung an Geschäftsprozessen[1065] kann als Kunden- und Marktorientierung angesehen werden, weil eine Anpassung der Unternehmens-IT an die Prozesse der Leistungserstellung für den Kunden bzw. Markt angestrebt wird. Die Leistun-

---

[1062] Beispielsweise wurde die Portabilität einer Unternehmens-IT gegenüber dem Wettbewerb als neutral eingestuft. In der Automobilindustrie könnte man sich allerdings vorstellen, dass Funktionen der Unternehmens-IT wie beispielsweise die Planung von Einsatzzeiten der Fahrzeugwartung in Werkstätten durch das Automobil während der Fahrt, auf der eine Fehlfunktion festgestellt wird, genutzt wird. Dem Fahrer wird daraufhin die nächste Werkstatt mit freien Kapazitäten mitgeteilt, und es wird ggf. direkt ein Termin mit der Werkstatt vereinbart. Die Portabilität von Teilen der Unternehmens-IT in Fahrzeuge des Herstellers wäre für dieses Produkt dann von Bedeutung und hätte Einfluss auf eine Bewertung.

[1063] Vgl. Aier, Schönherr /Flexibilisierung/ S. 33; Klement /Enterprise Architecture/ S. 177.

[1064] Vgl. Kapitel 3.2.2.7.2.

[1065] Vgl. Kapitel 2.2.5.2.

gen eines Unternehmens, die durch die Prozesse der Leistungserstellung erzeugt werden, betreffen den Kunden des Unternehmens in direkter oder indirekter Weise. Leistungen haben direkte Auswirkungen, wenn diese z. B. dem Markt als Dienstleistungen angeboten werden. Indirekte Auswirkungen weisen Leistungen auf, wenn z. B. die Qualität eines Produktes durch die Leistungserstellungsprozesse beeinflusst wird. Die Kundenzufriedenheit wird beeinflusst, sobald Prozesse der Leistungserstellung durch die Unternehmens-IT betroffen sind, unabhängig davon, ob diese den Markt direkt oder indirekt beeinflussen.

5.2.1.2 Auswirkungen auf Kundenzufriedenheit

Eine SOA wirkt in unterschiedlicher Weise auf die Kundenzufriedenheit: Durch die Beeinflussung von Lieferzeiten, durch kooperative Auftragsabwicklung, durch Erweiterung des Produktsortiments um innovative Produkte und durch eine Qualitätsbeeinflussung der Produkte.

Lieferzeiten

Wie in Kapitel 3.2.3.3.1 gezeigt ist die Integrationsreichweite einer SOA hoch ausgeprägt. Sie ermöglicht die Einbindung der IT-Systeme von Kunden über die Kommunikationsinfrastruktur. Dies führt dazu, dass kein Medienbruch bei der Übermittlung von Daten vorhanden ist und Zeit gespart werden kann.[1066] Im günstigsten Fall kann sogar eine direkte Interaktion zwischen den beteiligten Unternehmens-IT-Systemen stattfinden. Eine sehr hohe Verarbeitungsgeschwindigkeit kann hierdurch erreicht werden, weil der vollautomatische Ablauf von Prozessen unterstützt wird.[1067] Demzufolge sind zum einen die operativen Abwicklungen von Geschäftsbeziehungen, z. B. Bestell- oder Supportvorgänge, sehr schnell durchführbar; zum anderen muss der Kunde die ihm bekannte Umgebung der eigenen Unternehmens-IT nicht verlassen, um einen Geschäftsvorfall initiieren zu können. Solche Barrieren[1068] auf Kundenseite zur Anbahnung eines Ge-

---

[1066] Vgl. Scheckenbach /Geschäftsprozessintegration/ S. 7 nach Aier, Schönherr /Flexibilisierung/ S. 15.

[1067] Vgl. Klement /Enterprise Architecture/ S. 3-4.

[1068] Mellis beispielsweise unterscheidet Eintritts- und Wissensbarrieren, die Kunden daran hindern können, in Geschäftsbeziehung zu einem Unternehmen zu treten. Vgl. Mellis /Projektmanagement/ S. 33-35; Vgl. weiterführend Porter /Advantage/ S. 173 & S. 482-512.

schäftsvorfalls fallen weg, weil sich z. B. die Komplexität und Anzahl der (durch den Kunden) durchzuführenden Arbeitsschritte reduzieren lässt.[1069]

Durch die Reduktion der Zeit zur Anbahnung eines Geschäftsvorfalls verringert sich die Gesamtliefer- bzw. Gesamtdurchlaufzeit.[1070] Ein Beispiel hierfür ist, dass Vorgänge zur Datenerfassung in einer Unternehmens-IT direkt durch den Kunden angestoßen werden können und nicht durch einen Sachbearbeiter erfolgen müssen. Die entstehende Zeitersparnis kann dem Kunden direkt in Form von verkürzten Lieferzeiten angeboten werden.[1071]

Die Verringerung von Lieferzeiten kann bezüglich bestehender Produkte und Dienstleistungen und darüber hinaus auch auf der Ebene neu zu erstellender Produkte und Dienstleistungen betrachtet werden. Die Produktion und der Verkauf fast[1072] jedes Produktes und fast jeder Dienstleistung sind mit dem Einsatz einer Unternehmens-IT verbunden. Neue Produkte und Dienstleistungen müssen folglich durch die Unternehmens-IT unterstützt werden. Die Implementation einer SOA wird von Aier und Schönherr als Reaktion auf die Notwendigkeit verstanden, den Anforderungen einer sich schnell wandelnden betrieblichen Umgebung Rechnung zu tragen.[1073] Neue Produkte stellen eine Form der Reaktion auf sich wandelnde betriebliche Umgebungen dar. Eine hohe Wandlungsfähigkeit einer Unternehmens-IT ermöglicht eine schnelle Anpassung an Anforderungen und Bedürfnisse neuer Produkte. Die Wandlungsfähigkeit einer Unternehmens-IT auf Basis einer SOA wurde in Kapitel 3.2.4 sehr positiv bewertet, wodurch die Zeit,

---

[1069] Vgl. Gizanis, Legner, Österle /Kooperative Auftragsabwicklung/ S. 47-58. Gizanis, Legner und Österle stellen eine Fallstudie vor, in der ABB Robotics durch die Einbeziehung des Kunden in die Unternehmens-IT Einsparungen in Millionenhöhe bei gleichzeitiger Verbesserung des Servicelevels erreichen konnte. Vgl. Gizanis, Legner, Österle /Kooperative Auftragsabwicklung/ S. 55.

[1070] Vgl. Scheckenbach /Geschäftsprozessintegration/ S. 7 nach Aier, Schönherr /Flexibilisierung/ S. 15.

[1071] Vgl. Klement /Enterprise Architecture/ S. 3-4. In der Internetwelt kann man dies heute schon sehr gut beobachten. Unternehmen können Kunden sogar an arbeitsfreien Tagen vollständige Dienstleistungen, z. B. den Verkauf von Software, durchgängig von der Anbahnung über die Lieferung bis zur Bezahlung anbieten.

[1072] Produkte und Dienstleistungen kleiner Betriebe sind nicht unbedingt auf IT angewiesen. Die Produkte und Dienstleistungen der Unternehmen, die eine SOA einsetzen können, sind jedoch auf den Einsatz einer Unternehmens-IT angewiesen, um wettbewerbsfähig bleiben zu können. Begründen lässt sich dies damit, weil eine Abwicklung der Geschäftsvorfälle dieser Unternehmen ohne IT nicht mehr möglich ist.

[1073] Vgl. Carlson, Tyomkin /Good Design/ S. 13.

die ein Unternehmen benötigt, um neue Produkte und Dienstleistungen am Markt anbieten zu können, reduziert werden kann.[1074]

Kooperative Auftragsabwicklung

Eine kooperative Auftragsabwicklung[1075] bietet Kunden Vorteile und hat somit Auswirkungen auf die Kundenzufriedenheit. Eine kooperative Auftragsabwicklung ist ein Instrument der Gesamtunternehmensstrategie, deren Ausprägung für eine SOA in Kapitel 3.2.3.1 positiv bewertet wurde. Auswirkungen auf die Kundenzufriedenheit ergeben sich z. B. durch eine Verringerung von Beständen, durch gebündelte und aktuelle Informationen über Produkte und aktuelle Auftragszustände sowie durch die Verfügbarkeit von Produkten.

Bestände eines Produktes können bei Kunden durch die Verkürzung der Lieferzeiten reduziert werden. Produkte beim Kunden müssen i. d. R. in der Menge gelagert werden, die notwendig ist, um den Bedarf an diesen Produkten mindestens während ihrer Lieferzeit decken zu können.[1076] Verringert sich die Lieferzeit von Produkten, verringert sich auch der zu deckende Bedarf, wodurch Kunden ihre Lagerbestände und somit gebundenes Kapital reduzieren können.[1077] Kosteneinsparungen auf Seite des Kunden, die durch den Lieferanten bedingt sind, erhöhen die Kundenzufriedenheit.

Durch die Anbindung von Kunden an eine Unternehmens-IT können diese auf *aktuelle Informationen* zu angebotenen Produkten direkt zugreifen. Eine informationstechnische Integration aller an der Produktion beteiligten Unternehmensteile ermöglicht eine ganzheitliche Information für den Kunden. Dies kann z. B. Lieferzeiten, aktuelle Verarbei-

---

[1074] Vor allem im Bereich der Produkte, die zum großen Teil auf die Unterstützung durch eine Unternehmens-IT angewiesen sind, trifft dies zu. Beispiele hierfür sind Produkte des Geldmarktes, insbesondere von Versicherungen, bei denen neue Produkte schnell definiert sind und am Markt angeboten werden können, deren Abbildung in der Unternehmens-IT jedoch zu umfangreichen Erweiterungen führen.

[1075] Vgl. zu folgenden Absätzen Gizanis, Legner, Österle /Kooperative Auftragsabwicklung/ S. 47-58.

[1076] Vgl. Günther, Tempelmeier /Produktion/ S. 101-102. Anhand der von Günther und Tempelmeier vorgestellten Lagerhaltungspolitiken kann festgestellt werden, dass eine geringere Wiederbeschaffungszeit (aus Sicht des Kunden) dazu führen kann, dass Lagerbestände verringert werden. Verallgemeinert dargestellt kann konstruiert werden, dass Lagerbestände (solange Unsicherheit ignoriert wird) gegen Null gehen, wenn Lieferzeiten gegen Null gehen. Dies begründet sich darin, weil eine Bedarfsmenge bei einer Lieferzeit von Null direkt bei Bestellung vorliegen würde und eine Lagerhaltung dadurch entfallen kann. Günther, Tempelmeier /Produktion/ S. 261-273.

[1077] Vgl. zu weiteren Vorteilen verringerter Lagerbestände, beispielsweise zum Konzept der ‚Just in Time'-Anlieferung Günther, Tempelmeier /Produktion/ S. 297-301.

tungsstände von Bestellungen oder auch die Informationen umfassen, die den Kunden über Produktions- oder Lieferprobleme frühzeitig informieren.[1078]

Kunden, die direkt über die Unternehmens-IT mit dem Unternehmen interagieren, können direkt die *Verfügbarkeit* und antizipierte Lieferzeitpunkte von Produkten auf Basis aktueller Daten ermitteln. Solche Möglichkeiten erhöhen die Kundenzufriedenheit. Der Kunde kann benötigte Informationen zu den Zeitpunkten, an denen er sie benötigt, ohne Zeitverlust abfragen. Er ist nicht auf Interaktion mit Mitarbeitern des Unternehmens angewiesen oder muss sich nicht über andere Wege als der eigenen Unternehmens-IT informieren.

Innovative Produkte

Als Wettbewerbsstrategie kann auf eine SOA aufbauend eine ‚First-Mover-Strategie'[1079] verfolgt werden. Das Potential, die Kunden in die Unternehmens-IT zu integrieren, eröffnet die Möglichkeit, innovative Produkte und Dienstleistungen anbieten zu können.[1080] Beispiele hierfür sind die oben erwähnten Informationsdienste, die aktuelle Informationen an die Kunden in Echtzeit weitergeben können. In Kooperation mit Kunden können durch das Potential der Integration von Unternehmens-IT-Systemen spezialisierte Dienstleistungen für einzelne Kunden wie auch Kundengruppen angeboten bzw. speziell umgesetzt werden. Durch solche Mehrwertdienste können z. B. enge Kundenbeziehungen weiter vertieft werden.

Qualitätsverbesserung der Produkte

In der Literatur finden sich Argumentationen, dass eine aus verschiedenen Bausteinen zusammengesetzte Software Vorteile aufweist.[1081] Im Falle einer SOA liegen verschiedene Bausteine in Form von Services und der Kommunikationsinfrastruktur (sowie ih-

---

[1078] Es kann davon ausgegangen werden, dass die Kundenzufriedenheit geringer ist, wenn Probleme oder Verzögerungen auftreten. Allerdings wird der Kunde über Probleme und Verzögerungen so bald wie möglich informiert werden wollen, um dies in seiner Produktionsplanung beachten zu können und somit potentiell zufriedener sein.

[1079] Vgl. Porter /Advantage/ S. 186-189.

[1080] Vgl. Klement /Enterprise Architecture/ S. 3-4.

[1081] Vgl. zu den Vorteilen der Mehrfachverwendung Stahlknecht, Hasenkamp /Wirtschaftsinformatik/ S. 323.

rer Bestandteile) vor.[1082] Vorteile, die genannt werden, sind Steigerung der Qualität durch Wiederverwendung erprobter Bausteine und Komplexitätsreduktion durch separates Entwerfen, Entwickeln und Pflegen der einzelnen Services.[1083] Die Kundenzufriedenheit wird durch eine solche Qualitätsverbesserung der Unternehmens-IT direkt erhöht, wenn Kunden an eine Unternehmens-IT angebunden sind. Die Kundenzufriedenheit kann indirekt steigen, zum einen, weil interne Prozesse zur Bearbeitung von Kundenanfragen bzw. Aufträgen eine hohe Qualität aufweisen, z. B. durch eine hohe Verfügbarkeit[1084], hohe Robustheit[1085] oder hohe Erweiterbarkeit[1086]; zum anderen, weil Kundenanfragen auf Basis vollständiger und aktueller Daten durchgeführt werden können und folglich eine höhere Qualität aufweisen.

5.2.2 Mitarbeiterzufriedenheit

In Kapitel 5.1 wurde aufgezeigt, dass Wirtschaftlichkeitspotential unter anderem anhand der Mitarbeiterzufriedenheit diskutiert werden kann. Bei der Bewertung einer Unternehmens-IT kann die Mitarbeiterzufriedenheit bezüglich zwei Gruppen in einem Unternehmen beurteilt werden: Erstens wird die Gruppe der Mitarbeiter betrachtet, die eine Unternehmens-IT zur Erledigung ihrer Arbeit einsetzen muss; zum anderen ist die Gruppe der Mitarbeiter betroffen, die als Aufgabe den Aufbau und Betrieb der Unternehmens-IT ausübt.

Im folgenden Kapitel 5.2.2.1 wird zunächst argumentiert, warum Einfluss auf die Zufriedenheit der Mitarbeiter durch den Einsatz einer SOA besteht. Anschließend wird im Kapitel 5.2.2.2 dargestellt, welche Auswirkungen eine SOA auf die Mitarbeiterzufriedenheit der Anwender (Kapitel 5.2.2.2.1) und auf die Entwickler (Kapitel 5.2.2.2.2) einer Unternehmens-IT hat.

[1082] Vgl. Kapitel 2.2.4.

[1083] Vgl. zu diesem Absatz Rumbaugh u. a. /Modellieren/ S. 343-344 & S. 390; Reinertsen empfiehlt das Isolieren von Funktionen, die oft Veränderungen unterliegen, in kleine, preiswerte und einfach veränderbare Komponenten. Vgl. Reinertsen /System Architecture/ S. 22.

[1084] Vgl. Kapitel 3.2.2.4.1.

[1085] Vgl. Kapitel 3.2.2.4.1. Dies ist vor allem dann relevant, wenn Kunden direkt mit der Unternehmens-IT interagieren.

[1086] Vgl. Kapitel 3.2.2.2.1.

5.2.2.1 Einfluss einer SOA auf Mitarbeiterzufriedenheit

Die Mitarbeiterzufriedenheit der Anwender einer Unternehmens-IT wird von der Softwarearchitektur beeinflusst, weil Mitarbeiter auf die Verwendung der Unternehmens-IT zur Erfüllung ihrer Arbeit angewiesen sind. Eine Unternehmens-IT, die die Arbeit der Mitarbeiter erleichtert, wirkt sich positiv auf die Mitarbeiterzufriedenheit aus, ebenso der Grad der Unterstützung spezieller Anforderungen der Mitarbeiter. Unternehmen müssen z. B. die Anforderungen ihrer mobilen sowie ortsunabhängig eingesetzten Mitarbeiter[1087] erfüllen, um eine hohe Mitarbeiterzufriedenheit erreichen zu können.[1088] Solche Bedürfnisse betreffen auch die Unternehmens-IT – als unterstützendes System eines Unternehmens. Sie äußern sich darin, dass die Unternehmens-IT ihre Dienstleistungen überall dort anbieten kann, wo diese nachgefragt werden können. Beispielsweise kann dies der Heimarbeitsplatz eines Mitarbeiters sein oder der Laptop auf einem Flug zum Einsatzort bzw. Arbeitsplatz. Eine SOA hat als Softwarearchitektur Einfluss darauf, wie und ob solche allgemeinen Anforderungen der Mitarbeiter eines Unternehmens erfüllt werden können, weil sie Gestaltungsrichtlinien zum Aufbau der Unternehmens-IT darstellt.[1089]

Die Entwickler einer Unternehmens-IT sind direkt durch die Gestaltungsrichtlinien einer Softwarearchitektur betroffen. Der Aufbau der Software und – dadurch bedingt – der Prozess der Softwareentwicklung wird durch die Wahl einer Softwarearchitektur beeinträchtigt. Weil hierdurch die Arbeitsinhalte der Entwickler direkt betroffen sind, wirkt sich die Wahl einer Softwarearchitektur – also auch der Einsatz einer SOA – auf die Mitarbeiterzufriedenheit der Entwickler aus.

[1087] In die Kategorie der mobilen Angestellten fallen Personen, bei denen es zur Aufgabenerfüllung gehört, innerhalb kurzer Zeit an unterschiedlichen Orten ihren Aufgaben nachzukommen. Dies sind z. B. Angestellte im Vertretergeschäft oder Lieferanten. Ortsunabhängig eingesetzte Angestellte sind Personen, welche ihre Aufgaben i. d. R. nicht mobil erfüllen, die Aufgabenerfüllung allerdings ortsunabhängig geschehen kann. Zu diesem Personenkreis gehören z. B. Heimarbeiter und Unternehmensberater. In der heutigen Zeit fallen Personen auf der oberen Ebenen eines Unternehmens zwischen beide Kategorien. Beispielsweise müssen Manager häufig noch Zuhause oder auf Dienstreisen mobil ihren Aufgaben nachkommen.

[1088] Vgl. Oxman, Smith /Structual Change/ S. 79; In der Fallstudie von Moro und Lehner wurde Interoperabilität explizit als nicht-funktionale Anforderung aufgestellt. Vgl. Moro, Lehner /Reengineering/ S. 28.

[1089] Vgl. Klement /Enterprise Architecture/ S. 4.

5.2.2.2 Auswirkungen auf Mitarbeiterzufriedenheit

Eine Unternehmens-IT auf Basis einer SOA hat Einfluss auf die Mitarbeiterzufriedenheit sowohl für Anwender als auch Entwickler. Die Zufriedenheit von Mitarbeitern wird erhöht, wenn die Unternehmens-IT von den Software-Entwicklern so entwickelt und gewartet wird, dass diese die Anforderungen und Wünsche der Anwender genau abbildet. Eine solche Anforderungstreue unterstützt eine SOA durch die Gestaltungsrichtlinie, dass die Entwicklung und Strukturierung der Unternehmens-IT orientierend an Geschäftsprozessen durchgeführt werden soll.[1090] Darauf aufbauend kann auch eine Unternehmens-IT erstellt werden, die die jeweiligen prozessorientierten Organisationsprinzipien eines Unternehmens beachtet und deswegen als betriebswirtschaftliche Software geeignet ist.[1091] Auch beeinflusst eine erfolgreiche und reibungslose Softwareeinführung die Mitarbeiterzufriedenheit positiv, weil Gründe für Unzufriedenheit – wie z. B. Probleme in der Nutzung von Werkzeugen zur Arbeitsunterstützung oder Verzögerungen bei der Einführung notwendiger Unterstützungsinstrumente – vermieden werden können.

Auch lässt sich dies dadurch erklären, dass die Zufriedenheit von Mitarbeitern höher wird, wenn die Unternehmens-IT von den Mitarbeitern als ein modernes, innovatives und funktionierendes und nicht als ein veraltetes Hilfsmittel wahrgenommen wird. Eine SOA kann durchaus als moderne Softwarearchitektur aufgefasst werden. Dadurch könnte sie die Mitarbeiterzufriedenheit positiv beeinflussen. Andererseits kann die auf subjektiven Eindrücken beruhende Wirkung auf die Mitarbeiterzufriedenheit auch negativ ausfallen: Dann nämlich, wenn sich zeigt, dass die Unternehmens-IT auf Basis einer SOA z. B. weniger performant ist als die vorher eingesetzte[1092] oder wenn Sicherheitslücken entdeckt werden, die den Einsatz einschränken.[1093] Die Zufriedenheit der Mitarbeiter ist eine wichtige Voraussetzung, damit die Unternehmens-IT akzeptiert wird. Dies wiederum ist eine wichtige Voraussetzung für eine (erfolgreiche) Nutzung dieser.[1094]

[1090] Vgl. Kapitel 2.2.5.2.

[1091] Vgl. Frese /Organisation/ S. 132-133.

[1092] Vgl. Kapitel 3.2.2.5 in dem das Potential geringer Performanz identifiziert wurde.

[1093] Vgl. Kapitel 3.2.2.4 in dem Sicherheitsaspekte bewertet wurden.

[1094] Vgl. Macharzina /Unternehmensführung/ S. 670-671.

Diese aufgezeigte und durchaus spekulative Wirkung auf die Mitarbeiterzufriedenheit soll nicht als einziges Argument angeführt werden. Im Folgenden werden Auswirkungen auf Anwender (Kapitel 5.2.2.2.1) und auf Entwickler (Kapitel 5.2.2.2.2) im Detail dargestellt.

5.2.2.2.1 Auswirkungen auf Anwender einer Unternehmens-IT

Eine SOA wirkt in unterschiedlicher Weise auf die Mitarbeiterzufriedenheit der Anwender: Mittels unternehmensweit zugänglicher Daten und Prozesse, mittels Qualität durch Wiederverwendung und mittels Unterstützung der Arbeitsprozesse.

Daten und Prozesse

Wurde eine Unternehmens-IT auf Basis einer SOA aufgebaut, können Mitarbeiter (potentiell) auf eine integrierte Unternehmens-IT zurückgreifen. Der Integrationsgegenstand ist in diesem Fall die gesamte Unternehmens-IT. Folglich ist es möglich, dass alle Daten und Prozesse einheitlich im gesamten Unternehmen ab- und aufgerufen werden können.[1095] Eine Unternehmens-IT auf Basis einer SOA ist ein effektiveres Arbeitsmittel als einzelne nicht miteinander verbundene Systeme.[1096]

Die Integration aller IT-unterstützten Prozesse eines Unternehmens in einer Anwendung führt zu einer Verringerung von Medienbrüchen beim Wechsel zwischen Arbeitsaufgaben. Potentiell führt dies erstens zu einer Vermeidung von Fehlern, die durch inkorrekte Datenübertragung zwischen Anwendungen verursacht werden. Zweitens führt dies zu einer Beschleunigung von Abläufen, weil weniger Programme aufgerufen und bedient werden müssen. Drittens führt dies zu einer höheren Daten- und Prozessqualität, weil keine redundanten Daten[1097] in verschiedenen Programmen gehalten und Teilprozesse nicht redundant implementiert werden müssen.[1098]

[1095] Vgl. zum folgenden Absatz Kieser, Walgenbach /Organisation/ S. 403.

[1096] Vgl. Klement /Enterprise Architecture/ S. 4.

[1097] Dieser Aussage gegenüber stehen Aussagen, nach denen Redundanzen mitunter auch positiv sind und sogar notwendig sein können. Positive Aspekte von Redundanzen stellen beispielsweise Nonaka und Takeuchi vor (vgl. Nonaka, Takeuchi /Company/ S. 80-82). Sie beziehen sich in ihren Aussagen auf Daten in Form von Wissen. Diese intendierte Art der Redundanz bedeutet in diesem Fall, dass Mitarbeiter Wissen (identische Daten) internalisiert haben. Für die Softwareentwicklung kann dies z. B. die Architektur einer Software sein. Dadurch können sich Entwickler schnell und eindeutig austauschen, sowohl im direkten Gespräch als auch durch Dokumentation eines anderen Entwicklers. Datenredundanz kann durchaus auch notwendig sein: Will ein Unternehmen sichergehen, dass

Qualität

Die in Kapitel 3.2.2.6 positiv bewertete Wiederverwendung kann ebenfalls Auswirkungen auf die Mitarbeiterzufriedenheit ausüben. Eine SOA stellt eine aus eigenständigen Komponenten, sog. Services, zusammengesetzte Unternehmens-IT dar.[1099] Eine aus verschiedenen Bausteinen (z. B. Komponenten bzw. Services) zusammengesetzte Applikation hat den Vorteil, die Qualität einer Software zu erhöhen, indem erprobte Bausteine wiederverwendet werden.[1100] Eine potentiell höhere Qualität der Unternehmens-IT kann die Zufriedenheit der Mitarbeiter in den Fachabteilungen – also bei den Anwendern einer Unternehmens-IT – erhöhen.

Unterstützung des Arbeitsprozesses

Die Wandlungsfähigkeit einer Unternehmens-IT auf Basis einer SOA wurde in Kapitel 3.2.4 sehr positiv bewertet. Eine positive Wandlungsfähigkeit kann die Mitarbeiterzufriedenheit erhöhen. Sie ermöglicht unabhängige und Service-basierte Releasezyklen durch die lose Kopplung der Bestandteile der Unternehmens-IT. Dadurch können realisierbare neue Anforderungen schneller in Produktion gegeben werden als wenn die gesamte bzw. größere Teile einer Unternehmens-IT wegen einer Änderung aufeinander abgestimmt werden müssen.

Die Sicherheit einer Unternehmens-IT ist für Unternehmen und ihre Mitarbeiter von hoher Bedeutung. Die Sicherheit einer SOA wurde in Kapitel 3.2.2.1.1 negativ bewertet. Die Mitarbeiterzufriedenheit wird deswegen negativ durch eine Unternehmens-IT auf Basis einer SOA beeinflusst, weil die Unternehmens-IT gegen Unsicherheiten abgesichert werden muss. Solche Absicherungen können sich zum einen den Arbeitsfluss stören und verkomplizieren; zum anderen kann der Fall eintreten, dass eine Sicherheits-

---

Daten vor externen Wirkungen, die Datenverlust zur Folge haben können, geschützt sind, muss zwangsläufig mindestens eine 100 %-ig redundante Kopie aller Daten vorgehalten werden. Beide Arten von Datenredundanz werden jedoch nicht durch eine SOA beeinflusst und die Argumentationen im Hinblick auf positive Aspekte von Redundanzen stehen der hier dargestellten Argumentation nicht entgegen.

1098 Vgl. Scheckenbach /Geschäftsprozessintegration/ S. 7 (nach Aier, Schönherr /Flexibilisierung/ S. 15.)

1099 Vgl. Kapitel 2.2.4.1.

1100 Vgl. zu den Vorteilen der Mehrfachverwendung Stahlknecht, Hasenkamp /Einführung Wirtschaftsinformatik/ S. 323.

verletzung registriert wird, und die Unternehmens-IT aus diesem Grund nicht vollständig genutzt werden kann bzw. sogar kompromittiert wird.

Mobiles und ortsunabhängiges Arbeiten der Mitarbeiter ist heute für viele Unternehmen zu einer Notwendigkeit geworden.[1101] Die Verfügbarkeit, folglich auch die ortsungebundene Verfügbarkeit einer Unternehmens-IT auf Basis einer SOA wurde in Kapitel 3.2.2.4.1 neutral bewertet. Eine Erhöhung der Mitarbeiterzufriedenheit aufgrund der Erfüllung der Anforderungen mobilen und ortsunabhängigen Arbeitens kann daraufhin nicht festgestellt werden.

Das negativ bewertete Antwortzeitverhalten[1102] einer SOA wirkt sich nachteilig auf die Mitarbeiterzufriedenheit von Anwendern aus. Eine Unternehmens-IT verringert durch ein geringes Antwortzeitverhalten die Arbeitsgeschwindigkeit der Mitarbeiter. Wenn Mitarbeiter zu lange auf Reaktionen bei der Nutzung der Unternehmens-IT warten müssen, sinkt die Zufriedenheit mit der entsprechenden Anwendung.

5.2.2.2.2 Auswirkungen auf Software-Entwickler einer Unternehmens-IT

Eine SOA wirkt in unterschiedlicher Weise auf die Mitarbeiterzufriedenheit der Entwickler einer Unternehmens-IT: Durch Komplexitätsreduktion und durch die Unterstützung des Entwicklungsprozesses.

Komplexität in der Entwicklung

Die Benutzbarkeit einer SOA für Software-Entwickler wurde sehr positiv bewertet.[1103] Benutzbarkeit wirkt sich auf die Mitarbeiterzufriedenheit deswegen aus, weil Software-Entwickler die Unternehmens-IT als Arbeitsgegenstand handhaben müssen, z. B. die Entwicklung und der Betrieb der gesamten Unternehmens-IT. Eine SOA, die die Arbeit der Software-Entwickler vereinfacht, weil die Verständlichkeit der Komponenten hoch[1104] und die Entwicklung und Wartung von Schnittstellen unterstützt wird,[1105] führt zu einer höheren Mitarbeiterzufriedenheit der Software-Entwickler. Zudem unterstützt

---

[1101] Vgl. Kapitel 3.1.3.8.3.

[1102] Vgl. Kapitel 3.2.2.5.1.

[1103] Vgl. Kapitel 3.2.4.

[1104] Vgl. Kapitel 3.2.2.3.1.

eine SOA die Komplexitätsreduktion des Weiteren durch die Möglichkeit des separaten Entwerfens, Entwickelns und Pflegens einzelner Services.[1106]

Die Kommunikation über einen Service-Bus und Serviceschnittstellen, welche die Dienstleistung eines Services allen anderen Services anbietet, beinhaltet ebenso das Potential der Komplexitätsverringerung.[1107] Durch diese Gestaltungsrichtlinie einer SOA, inwieweit die Kommunikation unter Komponenten einer Unternehmens-IT strukturiert werden soll, wird vermieden, dass extrem viele Schnittstellen (neu-)entwickelt und gewartet werden müssen. Die Komplexität der Abhängigkeiten von Komponenten voneinander wird daraufhin reduziert.

Die Verständlichkeit, Betriebs- und auch Kommunikationsfähigkeit einer SOA wurden in Kapitel 3.2.2.3.1 neutral bewertet. Aus diesen Gründen sind keine weiteren Einflüsse auf die Mitarbeiterzufriedenheit von Software-Entwicklern ableitbar.

Softwareentwicklungsprozess

Der Softwareentwicklungsprozess umfasst den gesamten Lebenszyklus einer Unternehmens-IT – insbesondere die Entwicklung und Wartung. Verbesserungen wirken sich deswegen auf einen langen Zeitraum aus und können über die Zeit gerechnet potentiell eine starke Steigerung der Mitarbeiterzufriedenheit bewirken. Verbesserungen, die sich durch die Verwendung einer SOA ergeben können, sind erstens ein verringerter Zeitbedarf zur Umsetzung von Anforderungen durch Wiederverwendung. Zweitens unterstützt eine SOA durch ihre Gestaltungsprinzipien die Verwendung von Datenquellen, den Datenaustausch und die Datenverwendung innerhalb des gesamten Unternehmens.[1108] Drittens kann angenommen werden, dass die sehr positiv bewertete Benutzbarkeit[1109] die Entwicklung und Wartung einer Unternehmens-IT für Softwareentwicklung unkomplizierter gestaltet als wenn die Benutzbarkeit negativ bewertet wird. Viertens un-

---

[1105] Vgl. Kapitel 3.2.2.3.2 und Kapitel 3.2.2.3.1.

[1106] Vgl. Rumbaugh u. a. /Modellieren/ S. 343-344 & S. 390; Reinertsen empfiehlt das Isolieren von Funktionen, die oft Veränderungen unterliegen, in kleine, preiswerte und einfach veränderbare Komponenten. Vgl. Reinertsen /System Architecture to Profits/ S. 22.

[1107] Vgl. zum folgenden Absatz Aier, Schönherr /Flexibilisierung/ S. 11-14.

[1108] Vgl. Adam, McKendrick /Everything in Between/ S. 11. Adam und McKendrick sprechen hier von ‚internen Transaktionen'.

[1109] Vgl. Kapitel 3.2.4.

terstützt die sehr positive Wandlungsfähigkeit[1110] einer SOA die Wartung einer Unternehmens-IT. Diese Verbesserungen steigern die Mitarbeiterzufriedenheit der Software-Entwickler einer SOA.

5.2.3 Wettbewerbssituation

In Kapitel 5.1 wurde aufgezeigt, dass ein Wirtschaftlichkeitspotential unter anderem anhand von Wirkungen auf den Wettbewerb zwischen Unternehmen argumentiert werden kann. Im folgenden Kapitel 5.2.3.1 wird zunächst diskutiert, weshalb Einfluss auf die Wettbewerbssituation durch den Einsatz einer SOA besteht. Anschließend wird in Kapitel 5.2.3.2 der Einfluss einer SOA auf die Wettbewerbssituation dargestellt.

5.2.3.1 Einfluss einer SOA auf die Wettbewerbssituation

Von einer Unternehmens-IT wird die Fähigkeit erwartet, Wettbewerbsvorteile zu erhöhen und Wettbewerb zu revolutionieren.[1111] Belegt wird diese Erwartung durch eine Studie von Griffiths und Finlay. Die Autoren haben nachgewiesen, dass Wettbewerbsvorteile durch eine Unternehmens-IT realisiert werden konnten.[1112] Beispielhaft kann diese These durch die erfolgreichen Aktivitäten der ‚Billigflieger' im Luftverkehr bestätigt werden, welche ihre Vertriebsstruktur grundlegend neu (auf das Internet) ausgelegt haben. Daran lässt sich erkennen, dass es durch IT möglich ist, Wettbewerbsvorteile zu erlangen.

Eine Unternehmens-IT hat direkten Einfluss auf die Wettbewerbsfähigkeit eines Unternehmens.[1113] Sie unterstützt die Wertschöpfungsprozesse eines Unternehmens. Erreicht sie hierbei einen hohen Grad an effektiver und effizienter Unterstützung werden demzufolge direkt die erzielbaren Margen des erstellten Produktes beeinflusst. Ein Unternehmen kann bei einer positiven Entwicklung der Margen hinsichtlich einer steigenden

---

[1110] Vgl. Kapitel 3.2.4.

[1111] Vgl. Macharzina /Unternehmensführung/ S. 648-649; Empirische Ergebnisse zu dieser Erwartung liefern beispielsweise Griffiths und Finlay: die Autoren fanden heraus, dass 70 % der Teilnehmer einer Studie (65 US-Unternehmen des Finanz- und Einzelhandelsektores sowie des produzierenden Gewerbes) durch den Einsatz mindestens eines IT-Systems Wettbewerbsvorteile aufbauen oder aufholen konnten und diesen Vorteil im Schnitt sechs bis 18 Monate aufrecht erhalten konnten. Vgl. Griffiths, Finlay /Competitive Advantage/ S. 48-49.

[1112] Vgl. Griffiths, Finlay /Competitive Advantage/ S. 42-44.

[1113] Vgl. zum folgenden Absatz Porter /Advantage/ S. 164-172.

Unterstützung durch die Unternehmens-IT z. B. Einfluss auf den Wettbewerb ausüben, weil der Druck auf Konkurrenten durch Preissenkungen erhöht wird. Weiterhin besteht die Möglichkeit, andere wettbewerbswirksame Kriterien zu beeinflussen;[1114] beispielsweise durch die Erstellung von Eintrittsbarrieren, durch die Verkürzung von Produktionszeiten oder durch die Möglichkeit der Umsetzung individueller Kundenwünsche.

Unternehmen agieren heute oft in einer agilen Umgebung.[1115] Die Unternehmen sind gezwungen, ihr Verhalten und ihre Struktur den externen Bedingungen anzupassen. Als Teil eines Unternehmens ist eine Unternehmens-IT auch wegen Änderungen der externen Faktoren gezwungen, sich neuen Anforderungen und Strukturen anzupassen. Gegenwärtig soll das Architekturdesign die Verantwortung übernehmen, sowohl die Unternehmensperformanz zu erhöhen als auch eine hohe Flexibilität zuzulassen.[1116] Durch festgelegte Gestaltungsrichtlinien hat eine Softwarearchitektur durchaus Einfluss auf die Fähigkeit, Anpassungen vornehmen zu können.

5.2.3.2 Auswirkungen auf die Wettbewerbssituation

Wettbewerbsvorteile beruhen auf einer mit Wettbewerbern verglichenen überlegenen Ausnutzung von Ressourcen eines Unternehmens.[1117] Eine Wettbewerbssituation kann dadurch verbessert werden, dass man Vorteile gegenüber den Wettbewerbern realisiert oder die Vorteile von Wettbewerbern aufholt und somit zunichte macht. Nach Krüger stehen Unternehmen mittlerweile permanent unter dem Druck, sich zu wandeln um wettbewerbsfähig zu bleiben.[1118] Hauptgründe sieht er in Globalisierungsbestrebungen, Änderungen im Bereich der Arbeitskräfte (z. B. durch demographischen Wandel und durch Professionalisierung ehemals unterentwickelter Länder) und durch technische Evolution. Grundlegend können Verbesserungen der Wettbewerbssituation durch zwei Vorgehen erreicht werden:[1119]

[1114] Vgl. Porter /Advantage/ S. 172-176.

[1115] Vgl. Mellis /Turbulent Times/ S. 278.

[1116] Vgl. MacCormack, Verganti, Iansiti /Products/ S. 145.

[1117] Vgl. zum folgenden Absatz Griffiths, Finlay /Competitive Advantage/ S. 33.

[1118] Vgl. Krüger /Management/ S. 228. Bestätigt wird er durch weitere Autoren, beispielsweise Lewin, Hunter /Information Technology/ S. 255.

[1119] Vgl. zum folgenden Absatz Kieser, Walgenbach /Organisation/ S. 423-425; Griffiths, Finlay /Competitive Advantage/ S. 32-33; Baumöl, Winter /Qualifikation/ S. 46-47.

1. Ein Unternehmen passt sich an das Umfeld an.
   Dies wird z. B. erreicht, indem Unternehmen Kundenbedürfnisse besser befriedigen als Wettbewerber.

2. Das Umfeld wird an das Unternehmen angepasst.
   In diesem Fall sind Unternehmen bestrebt, die Bestandteile des Umfelds so zu beeinflussen, dass sie die eigenen Aufgaben besser[1120] erfüllen können. Beispielsweise kann Zulieferern empfohlen werden, in direkter Nachbarschaft zum eigenen Produktionsstandort zu produzieren bzw. Produkte zu lagern. Auch können durch Lobbyarbeit gesetzliche Rahmenbedingungen beeinflusst werden.

Beide Möglichkeiten zur Steigerung der Verbesserung der Wettbewerbsposition werden im Folgenden untersucht.

5.2.3.2.1 Auswirkungen durch Anpassung des Unternehmens

Unternehmen müssen in der Lage sein, sich pro-aktiv an Veränderungen der Umwelt anpassen zu können.[1121] Die Fähigkeit einer möglichst schnellen Anpassung wird als Wettbewerbsvorteil angesehen. Die Wandlungsfähigkeit einer SOA wurde in Kapitel 3.2.4 sehr positiv bewertet. Deswegen können Unternehmen, die eine SOA einsetzen, durch eine schnelle Reaktionsfähigkeit Wettbewerbsvorteile realisieren. Je höher der Anteil der Unternehmens-IT an allen Investitionen ist, um Änderungen durchführen zu können, desto höher sind die Wettbewerbsvorteile.

Automatisierung

Die Automatisierung von betrieblichen Aufgaben gewährleistet die Realisierung von Wettbewerbsvorteilen.[1122] Wird eine SOA zur Implementation einer integrierten Unternehmens-IT verwendet, können einzelne Schritte eines Geschäftsprozesses automatisiert

---

[1120] ‚Besser' ist in diesem Kontext bewusst umfassend gehalten, da sich potentielle Änderungen auf extrem viele Bereiche des Umfelds beziehen können. ‚Besser' wird andererseits typischerweise ‚in kürzerer Zeit', ‚mit geringerem Mitteleinsatz' oder auch ‚in höherer Qualität' im Vergleich zu der Situation vor entsprechenden Änderungen bedeuten.

[1121] Vgl. zum folgenden Absatz Gronau /Wandlungsfähige lnformationssystemarchitekturen/ S. 43-44 & S. 274; Anpassung des Unternehmens an sein Umfeld ist eine von zwei Vorgehensweisen, um Verbesserungen der Wettbewerbsposition zu erreichen. Vgl. Kieser, Walgenbach /Organisation/ S. 423-425; Griffiths, Finlay /Competitive Advantage/ S. 32-33; Baumöl, Winter /Qualifikation/ S. 46-47.

[1122] Vgl. Festl, Sinz /Wirtschaftsinformatik/ S. 189.

durchgeführt werden. Diesen Vorteil bieten alle Softwarearchitekturen, die eine Integration einer Unternehmens-IT ermöglichen. Die SOA wurde in Kapitel 3.2.3 bezüglich der Nachhaltigkeit und bezüglich der Integriertheit sehr positiv bewertet. Durch diese Ausprägungen kann eine SOA zum einen Unternehmensstrukturen abbilden und zum anderen eine Integration aller Bestandteile einer Unternehmens-IT herbeiführen. Beide Eigenschaften unterstützen die Automatisierung betrieblicher Aufgaben. Erstens, weil die Unternehmens-IT an die Strukturen der anfallenden Aufgaben und an die Struktur der durchführenden Unternehmenseinheiten angepasst ist, zweitens, weil sie alle zur Abarbeitung notwendigen Teile der Unternehmens-IT vereinen kann.

Die Automatisierung kann auch über die Grenzen eines Unternehmens hinaus zur Realisierung von Wettbewerbsvorteilen führen. Werden die IT-Systeme der Lieferanten und Kunden an die Unternehmens-IT angebunden, kann eine Automatisierung über Unternehmensgrenzen durchgeführt werden. Eine SOA bietet die Möglichkeit, Lieferanten und Kunden an die Unternehmens-IT anzubinden, und sie wird bezüglich der Integrationsreichweite positiv bewertet.

Vorhandene Wettbewerbsvorteile

Griffiths und Finlay haben Faktoren identifiziert, die einen schon bestehenden Wettbewerbsvorteil in einen langfristig[1123] anhaltenden transformieren.[1124] Diese Faktoren sind: Eine verbesserte interne und externe Kommunikation, erhöhter Level der Kundenzufriedenheit, Unterstützung des organisatorischen Lernens, Verbesserung der organisatorischen Flexibilität sowie innovative Nutzung der Informationstechnik.[1125] Die Unterstützung organisatorischen Lernens ist zu allgemein formuliert, um konkrete Fähigkeiten zur Beeinflussung des Wettbewerbsvorteils durch eine Softwarearchitektur identifizieren zu können. Eine innovative Nutzung der Informationstechnik ist durch eine Reduktion von Abhängigkeiten beim Einsatz einer SOA schnell möglich.[1126] Innovationen unterstützen eine SOA jedenfalls nur im Sinne von komplementären Produkten, die zum großen Teil auf der Existenz der Unternehmens-IT beruhen. Die Erhöhung

---

[1123] Griffiths und Finlay definieren langfristig als einen Zeitraum von mehr als 18 Monaten, vgl. Griffiths, Finlay /Competitive Advantage/ S. 45-46.

[1124] Vgl. Griffiths, Finlay /Competitive Advantage/ S. 46-48.

[1125] Vgl. Griffiths, Finlay /Competitive Advantage/ S. 45-48.

[1126] Vgl. META Group /Approaches/ S. 2 und Tannhäuser, Umek /Architekturmanagement/ S. 68.

der Kundenzufriedenheit wurde in Kapitel 5.2.1.2 dargestellt. Organisatorische Flexibilität wurde oben angesprochen. Eine verbesserte interne und externe Kommunikation kann durch eine integrierte Unternehmens-IT, die über die Grenzen des Unternehmens hinaus kommunizieren und agieren kann,[1127] erreicht werden. Die Integriertheit einer SOA wurde in Kapitel 3.2.4 sehr positiv bewertet. Die Möglichkeit einer hohen Integrationsreichweite eignet sich speziell zur Verbesserung der externen Kommunikation.

Anpassung der Wertschöpfungskette

Die Anpassungsmöglichkeiten eines Unternehmens kann durch eine Softwarearchitektur beeinflusst werden.[1128] Anpassungsmöglichkeiten wirken auf die Wettbewerbssituation, weil Sie die Wertschöpfungsketten (die Leistungserstellungsprozesse) eines Unternehmens betreffen. Änderungen der Wertschöpfungsketten können in vier Dimensionen unterteilt werden:[1129] Produkt- & Kundenreichweite (Segment scope), Fertigungstiefe (Vertical scope), geographische Reichweite (Geographic scope) und den Wettbewerbsmarkt (Industry scope). Änderungen der Wertschöpfungsketten wirken sich dadurch auf die Wettbewerbssituation eines Unternehmens aus.

Die *Produktreichweite* kann durch die Erweiterung des Leistungsangebots mit komplementären Produkten und Dienstleistungen expandiert werden,[1130] weil mittels einer SOA komplementäre Dienstleistungen gegenüber Kunden ermöglicht werden.[1131] Die *Kundenreichweite* wird durch den Einsatz einer SOA nicht beeinflusst. Zwar ist es möglich Kunden, z. B. besser zu informieren, jedoch werden durch den Einsatz einer Softwarearchitektur keine neuen Kunden angesprochen. Eine SOA kann die Kundenreichweite nur indirekt beeinflussen, indem diese Änderungen im Unternehmen, z. B. die Erstellung neuer Produkte, unterstützt. Infolgedessen kann eine neue Kundengruppe angesprochen werden.

Die *Fertigungstiefe* kann durch eine Unternehmens-IT beeinflusst werden, weil sie die Bestimmung von Partnern zur effizienten Erfüllung eines Auftrags auf Basis von Echt-

---

[1127] Vgl. Kapitel 3.2.3.1.1.

[1128] Vgl. Kapitel 3.1.3.3.3.

[1129] Vgl. Porter /Advantage/ S. 53-57.

[1130] Vgl. Gizanis, Legner, Österle /Kooperative Auftragsabwicklung/ S. 47.

[1131] Vgl. Kapitel 5.2.1.2.

zeitinformationen ermöglicht.[1132] Dazu müssen Informationen für alle an der Wertschöpfungskette beteiligten Unternehmen als auch für alle Unternehmen, die potentiell zur Erfüllung eines Auftrages eingesetzt werden können, zu einem Zeitpunkt vorliegen. Eine Unternehmens-IT muss deswegen die Kommunikation zu externen Unternehmen ermöglichen. Eine SOA ist dazu in der Lage.[1133] Die Einbindung von Unternehmen zum Informationsaustausch und zur Auftragsverarbeitung bietet auch die Möglichkeit, eine bessere Kapazitäts- und Ressourceneinsatzplanung infolge zeitnaher Auftragsinformation durchführen zu können.[1134]

Die *geographische Reichweite* wird durch eine SOA auch positiv beeinflusst, weil diese bezüglich der Integriertheit in Kapitel 3.2.4 sehr positiv verwertet wurde. Mittels einer SOA ist es möglich, externe Unternehmen und Kunden unabhängig von ihrem Standort einzubinden. Gleichzeitig kann sie einen besseren Zugang zu Faktor- und Absatzmärkten bieten. Ein weiterer Aspekt ist die Portabilität. Diese wurde in Kapitel 3.2.4 jedoch sehr negativ bewertet. Dadurch ist es schlecht möglich, die Dienstleistungen der Unternehmens-IT beliebig an wechselnden Orten verwenden zu können. Einzig die Portabilität von Daten und Funktionen wurde positiv bewertet,[1135] da Daten und Funktionen in Echtzeit zur Verfügung gestellt werden könnten – wenn dies trotz der Schwierigkeiten, die die negative Bewertung bedingen, realisiert wird.

Der *Wettbewerbsmarkt* kann durch zwei Faktoren beeinflusst werden: nämlich Veränderungen des Marktes und das Verhaltens eines Unternehmens am Markt. Der Markt ändert sich z. B., wenn Wettbewerber aus dem Markt gedrängt oder wenn neue Produkte angeboten werden. Die Verdrängung von Wettbewerbern hat Einfluss auf eine Unternehmens-IT, wenn die Wettbewerber übernommen werden.[1136] Die Aktivitäten der beiden ehemaligen Konkurrenten müssen aufeinander abgestimmt werden, und ggf. werden die Unternehmen vollständig integriert. In beiden Fällen muss die Unternehmens-IT angepasst werden. In solchen Situationen können Änderungen die Struktur und Strategie des Unternehmens sowie neue oder geänderte Geschäftsprozesse umfassen. Der erste Fall, die gesamte Novellierung, wird durch eine SOA unterstützt, weil die Wand-

---

[1132] Vgl. Gizanis, Legner, Österle /Kooperative Auftragsabwicklung/ S. 47.

[1133] Vgl. Kapitel 5.2.4.

[1134] Vgl. Gizanis, Legner, Österle /Kooperative Auftragsabwicklung/ S. 47.

[1135] Vgl. Kapitel 3.2.2.7.2.

lungsfähigkeit, die Integriertheit, die Strategieunterstützung und auch die Nachhaltigkeit sehr positiv bewertet wurden.[1137] Der zweite Fall, das Anbieten neuer Produkte, unterstützt eine Unternehmens-IT direkt, wenn Produkte angeboten werden können, die zum großen Teil auf der Unternehmens-IT als Produktionsfaktor beruhen.[1138] Indirekten Einfluss hat eine Unternehmens-IT, wenn eine schnelle Marktpräsenz erzielt werden kann,[1139] und folglich Wettbewerbsvorteile durch so genannte „First-Mover"[1140]-Vorteile erreicht werden können.[1141] Eine SOA ermöglicht folglich betreffend eines neuen Produktes die Verfolgung einer First-Mover-Strategie.

Wettbewerbsbedingte Notwendigkeit

Unternehmen können einerseits aus ihrer Unternehmens-IT strategische Wettbewerbsvorteile schöpfen.[1142] Die wettbewerbsrelevanten Kriterien bezüglich einer SOA wurden oben dargestellt. Andererseits können Investitionen in eine Unternehmens-IT eine wettbewerbsbedingte Notwendigkeit werden.[1143] Beispielsweise, wenn bestimmte Dienstleistungen oder komplementäre Produkte für die angebotenen Produkte selbstverständlich geworden sind. Wenn Aktivitäten zur Erweiterung einer Unternehmens-IT sehr langsam und in geringem Ausmaß geschehen, droht die Unsicherheit, dass ein Unternehmen den Anschluss an den Markt verliert und aus diesem gedrängt wird.[1144] Durch eine gute Wandlungsfähigkeit kann vermieden werden, dass technische Hindernisse zu solchen Situationen führen. Eine SOA wurde bezüglich der Wandlungsfähigkeit sehr positiv bewertet.[1145]

[1136] Der Fall, dass ein Wettbewerber aus dem Markt ausscheidet, betrifft die Unternehmens-IT nicht.

[1137] Vgl. Kapitel 3.2.4.

[1138] Als Beispiel kann der Versicherungssektor genannt werden. Neue Produkte müssen hier in der Regel durch die Unternehmens-IT erfasst und mit den entsprechenden Konditionen berechnet und verwaltet werden können. Eine weitere Produktion findet hier nicht statt.

[1139] Vgl. Bass u. a. /Software architecture/ S. 95.

[1140] Vgl. Porter /Advantage/ S. 186-189.

[1141] Strebt ein Unternehmen eine ‚First-Mover'-Strategie an, so sei an dieser Stelle darauf hingewiesen, dass auch ‚First-Mover'-Nachteile existieren.

[1142] Vgl. Lucas /Information Technology/ S. 109-116.

[1143] Vgl. Lucas /Information Technology/ S. 116-124.

[1144] Karch u. a. /SAP/ S. 25.

[1145] Vgl. Kapitel 3.2.4.

Verarbeitungsgeschwindigkeit

Werden Produkte am Markt angeboten, die zum großen Teil auf Dienstleistungen der Unternehmens-IT beruhen und die durch Kunden direkt verwendet werden können, ist die Effizienz einer Unternehmens-IT wichtig. Die Verarbeitungsgeschwindigkeit eines Produktes wird dann zu einem Qualitätskriterium. Ein Abnehmer findet z. B. keinen Mehrwert in der Möglichkeit, das aktuelle Lieferdatum für seine Bestellung zu erfahren, wenn seiner Empfindung nach die Dauer der Ermittlung der Daten inakzeptabel ist.[1146] Die Effizienz einer SOA wurde in Kapitel 3.2.4 negativ bewertet und bietet deswegen keine Wettbewerbsvorteile für solche Produkte.

Das Instrument der Kostensenkung[1147] ist ein weiterer wichtiger Punkt, der den Wettbewerb beeinflusst. Eine Betrachtung von Potentialen zur Kostensenkung wird gesondert im Kapitel 5.2.5 vorgenommen.

5.2.3.2.2 Auswirkungen durch Anpassung des Umfelds

Der zweite Faktor, um den Wettbewerb zu beeinflussen, ist das Umfeld eines Unternehmens so zu verändern, dass dieses den Anforderungen und Eigenschaften des Unternehmens besser gerecht wird.[1148] Die einzige – indirekte – Möglichkeit einer Unternehmens-IT, auf das Umfeld eines Unternehmens Einfluss auszuüben, besteht über die Integrationsfähigkeit einer Softwarearchitektur, weil dies die einzige Verbindung des Qualitätsmodells ist, die zum Umfeld eines Unternehmens Kontakt hat.

Wenn ein Unternehmen mittels seiner Unternehmens-IT Informations- und Interaktionsmöglichkeiten über die Unternehmensgrenzen hinaus anbietet, kann dies zu Anpassungen des Marktes führen bzw. können solche durchgesetzt werden. Beispielsweise können Unternehmen mit einer hohen Marktmacht gegenüber Faktormärkten durchsetzten, dass Lieferanten bestimmte Teile der Unternehmens-IT verwenden müssen, um weiterhin Aufträge erhalten zu können. Infolgedessen kann ein Unternehmen z. B. ad-

---

[1146] Vgl. hierzu beispielsweise Argumentation bzgl. der Seitenaufbaugeschwindigkeit von Internetseiten Herzwurm, Trittmann /Qualität/ S. 68.

[1147] Vgl. Macharzina /Unternehmensführung/ S. 650.

[1148] Vgl. Kieser, Walgenbach /Organisation/ S. 423-425; Griffiths, Finlay /Competitive Advantage/ S. 32-33; Baumöl, Winter /Qualifikation/ S. 46-47.

ministrative und kommunikative Aktivitäten mit Lieferanten effizienter und teilweise vollautomatisch durchführen.

5.2.4 Beziehungen zu Geschäftspartnern

In Kapitel 5.1 wurde aufgezeigt, dass das Wirtschaftlichkeitspotential unter anderem anhand der Beziehung zu Geschäftspartnern argumentiert werden kann. Im folgenden Kapitel 5.2.4.1 wird zunächst argumentiert, aus welchem Grund Einfluss auf die Beziehung zu Geschäftspartnern durch den Einsatz einer SOA besteht. Anschließend wird in Kapitel 5.2.4.2 der Einfluss einer SOA auf die Beziehung zu Geschäftspartnern dargestellt.

5.2.4.1 Einfluss einer SOA auf Beziehungen zu Geschäftspartnern

IT stellt eine Option dar, die den Gestaltungsspielraum des Organisationsgestalters erweitert[1149] und dient zur Unterstützung der Organisation. Eine Organisation muss mit verschiedenen Geschäftspartnern zusammenarbeiten. Beispiele für Geschäftspartner sind Lieferanten, Dienstleistungsanbieter, Ämter und Kunden.[1150] Weil der Umgang mit und die Auswahl von Geschäftspartnern ein Gegenstand der Organisationsgestaltung ist, besitzt die IT, innerhalb eines Unternehmens in Form einer Unternehmens-IT, die Eigenschaft, Einfluss auf die Beziehung mit Geschäftspartnern ausüben zu können. Da eine Unternehmens-IT anhand der Gestaltungsrichtlinien einer Softwarearchitektur aufgebaut wird, besitzt eine Softwarearchitektur, folglich auch eine SOA, Einflussmöglichkeiten auf die Beziehungen zu Geschäftspartnern. In der Reise- und Luftfahrtbranche z. B. ist eine effiziente Zusammenarbeit mit Geschäftspartnern über IT-Systeme extrem wichtig geworden, um am Markt bestehen zu können.[1151]

5.2.4.2 Auswirkungen auf Beziehungen zu Geschäftspartnern

Die Optionen zur Gestaltung der Beziehungen zu Geschäftspartnern werden anhand von zwei Zielen gestaltet:

[1149] Vgl. Frese /Organisation/ S. 142-144.

[1150] Vgl. Kapitel 5.1; Für die Gruppe der ‚Kunden' wurde eine Bewertung schon in Kapitel 5.2.1 anhand der Kundenzufriedenheit vorgenommen.

[1151] Vgl. Buhalis /eAirlines/ S. 816-818.

1. Verbesserung der operativen Zusammenarbeit mit Geschäftspartnern[1152] und
2. Effizientere Gestaltung der Geschäftsprozesse mit Geschäftspartnern.[1153]

Operative Zusammenarbeit

Die Verbesserung der operativen Zusammenarbeit mit Geschäftspartnern kann durch eine Unternehmens-IT erreicht werden, indem die Kommunikation mit Geschäftspartnern direkt und auf Basis korrekter Daten durchgeführt werden kann. Eine weitere Verbesserung kann vor allem durch eine effizientere Gestaltung der Kanäle zum Informationsaustausch erfolgen.[1154] Besitzt eine Softwarearchitektur die Fähigkeit, andere Unternehmens-IT-Systeme einbinden zu können, ist dies die effizienteste Art der Kommunikation, weil Informationen vollautomatisch ausgetauscht werden können. Darüber hinaus ergeben sich weitere Vorteile für ein Unternehmen, wenn die Unternehmens-IT die Ausführung von Teilen der Unternehmens-IT bei Geschäftspartnern oder auf Geräten, die mobil verwendet werden können, ermöglicht.[1155] Beispiele sind die ortsunabhängige Geschäftsabwicklung bei Geschäftspartnern/auf Messen und die Verfügbarkeit von aktuellen Daten und Prozessen beim Geschäftspartner. Beides führt zu einer effizienten und zeitsparenden Ressourcennutzung, weil Geschäftsabschlüsse bei dem oder durch den Geschäftspartner ausgeführt werden können. Weil eine SOA bezüglich des Integrationsgegenstandes positiv bewertet wurde, kann sie die Anforderung der Integration von anderen Unternehmens-IT-Systemen und auch von mobilen Endgeräten erfüllen.

[1152] Vgl. Klement /Enterprise Architecture/ S. 5.

[1153] Lewin, Hunter /Information Technology/ S. 255-256; Klement /Enterprise Architecture/ S. 5; Adam und McKendrick sehen in der Möglichkeit, externe Geschäftspartner in eine Unternehmens-IT einzubinden, Potential, weitere und neue Geschäfte mit diesem Geschäftspartner durchführen zu können (‚Business enabler'). Vgl. Adam, McKendrick /Everything in Between/ S. 11-12.

[1154] Weitere Aktivitäten auf operativer Ebene, wie z. B. den Transport von Produkten, kann eine Unternehmens-IT nicht übernehmen, und diese werden daher an dieser Stelle nicht untersucht. Es wird darauf hingewiesen, dass eine Unternehmens-IT diese Tätigkeiten unterstützten kann, indem optimale Lieferzeiten und Wege, durchaus auch in Interaktion zweier Unternehmens-IT-Systeme, ermittelt werden. Diese Aktivitäten allerdings werden als Informationsaustausch angesehen und sind somit durch die Betrachtung abgedeckt.

[1155] Vgl. zum folgenden Absatz Hansen, Neumann /Wirtschaftsinformatik/ S. 539-540.

Gestaltung der Geschäftsprozesse

Eine effizientere Gestaltung der Geschäftsprozesse mit Geschäftspartnern kann dadurch erfolgen, wenn Geschäftsprozesse an der Stelle durchgeführt werden, an denen diese auch angestoßen werden. Eine Unternehmens-IT kann dies unterstützen, indem sie Teile ihrer intern angebotenen Dienstleistungen auch extern verfügbar macht. Eine hohe Portabilität ist vorteilhaft, damit eine Unternehmens-IT Teile ihrer Funktionalität extern verfügbar machen kann.[1156] Abgesehen davon, dass die Portabilität von Software auch stark von der Portabilität der verwendeten Technik abhängt,[1157] wurde eine SOA in Kapitel 3.2.4 bezüglich der Portabilität sehr negativ bewertet und unterstützt dieses Wirtschaftlichkeitskriterium folglich schlecht.

Betrachtet man ein Unternehmen als Ganzes ergibt sich durch die Einbindung von Geschäftspartnern in die Unternehmens-IT ein weiteres Verbesserungspotential. Durch die Möglichkeit, aktuelle Daten aller Geschäftspartner einsehen zu können, ergibt sich eine bessere Planungs- und Steuerungssituation über alle Geschäftsprozesse hinweg.[1158] Beispielsweise können die Geschäftspartner identifiziert werden, die die Anforderungen eines Auftrages am besten erfüllen können. Werden alle Geschäftspartner in eine Unternehmens-IT integriert, muss diese andererseits in der Lage sein, alle Nutzergruppen entsprechend bedienen zu können. Eine SOA wurde in Kapitel 3.2.2.5 jedoch bezüglich der Effizienz (insbesondere durch das Antwortzeitverhalten) negativ beurteilt.

Des Weiteren unterliegen Geschäftsbeziehungen oft Änderungen. Eine Unternehmens-IT muss als Grundvoraussetzung in der Lage sein, Änderungen schnell umsetzen zu können. Bezüglich der Wandlungsfähigkeit wurde eine SOA sehr positiv bewertet, weshalb diese Anforderung erfüllt werden kann.[1159]

5.2.5 Interne Wirkungen

In Kapitel 5.1 wurde aufgezeigt, dass das Wirtschaftlichkeitspotential unter anderem anhand der internen Wirkungen argumentiert werden kann. Im Kapitel 4 wurde mit dem

---

[1156] Vgl. Raasch /Systementwicklung/ S. 36.

[1157] Dieser technische Aspekt wird jedoch, wie in Kapitel 1.3.3 dargestellt, in dieser Arbeit nicht bewertet.

[1158] Vgl. Gizanis, Legner, Österle /Kooperative Auftragsabwicklung/ S. 47.

[1159] Vgl. Kapitel 3.2.4.

Kostenpotential untersucht, welche Kosten der Einsatz einer SOA im Speziellen verursachen kann. Bei der Analyse der internen Wirkungen dagegen wird aufgezeigt, welche Kosten durch den Einsatz einer SOA reduziert werden können.

Kostensenkung mit einer SOA können durch direkte und indirekte Wirkungen erreicht werden. Direkte Kosten stehen direkt mit der Entwicklung, dem Betrieb und der Verwendung einer Unternehmens-IT in Zusammenhang. Kostenvorteile in diesem Bereich lassen sich durch die Verwendung von Standardprodukten erzielen, bei denen man unter verschiedenen Herstellern den insgesamt betrachtet kostengünstigsten auswählen kann. Indirekte Kosteneinsparungen lassen sich erreichen, weil ein Potential besteht, die zukünftigen Umsetzungen von Änderungsanforderungen in kürzerer Zeit und höherer Qualität zu erzielen. In der Literatur findet sich viel Optimismus, welche Kosten durch den Einsatz einer SOA verringert werden können. Darunter fallen z. B. Ausführungen, die die Wiederverwendbarkeit von Services positiv hervorheben[1160] oder eine Qualitätssteigerung[1161] durch eine SOA feststellen und dadurch Kostenvorteile entstehen. Diese und weitere Argumentationen werden in diesem Kapitel untersucht.

Im folgenden Kapitel 5.2.5.1 wird zunächst argumentiert, warum Einfluss auf interne Wirkungen durch den Einsatz einer SOA besteht. Anschließend wird in Kapitel 5.2.5.2 der Einfluss einer SOA auf interne Wirkungen dargestellt.

5.2.5.1 Einfluss einer SOA auf interne Wirkungen

Eine SOA bringt Gestaltungsrichtlinien, die festlegen, wie eine Unternehmens-IT strukturiert und entwickelt werden soll. Diese Gestaltungsrichtlinien haben somit Einfluss auf die Arbeitsinhalte (z. B. Prozesse) und Arbeitsergebnisse (z. B. Services) der Software-Entwickler einer Unternehmens-IT. Sowohl die Abarbeitung von Arbeitsinhalten als auch die Erstellung von Arbeitsergebnissen verursachen Kosten. Sowohl Arbeitsinhalte als auch Arbeitsergebnisse zwischen Softwarearchitekturen können unterschiedlich ausgeprägt sein. Wenn weitere Einflussfaktoren konstant gehalten werden, sind die entstehenden Kosten einer Unternehmens-IT ungleich ausgeprägt. Aus diesem Grund weist eine SOA Eigenschaften auf, die sich auf entstehende Kosten auswirken.

[1160] Vgl. Bass u. a. /Software architecture/ S. 106; Bendzulla, Stülpnagel /Kooperative Entwicklung/ S. 53-54; Lublinsky, Tyomkin /Dissecting SOA/ S. 56; Natis /SOA/ S. 25; Adam, McKendrick /Everything in Between/ S. 21

5.2.5.2 Auswirkungen auf interne Wirkungen

Eine Beeinflussung von Kosten kann auf drei Kostenposten zurückgeführt werden. Kosten der Erstellung und des Betriebs einer Unternehmens-IT können auf Personalkosten (1) und auf Ausgaben für verwendete Produkte (2) zur Erstellung und Wartung einer Unternehmens-IT zurückgeführt werden.[1162] Die Kosten der Verwendung einer Unternehmens-IT werden dagegen nur auf Personalkosten (3) zurückgeführt. Dies erklärt sich dadurch, weil die Kosten von Produkten, die während der Verwendung einer Unternehmens-IT anfallen, niedriger sind als die entstehenden Kosten für Produkte, die bei der Entwicklung verwendet werden. Dies begründet sich darin, weil im Einsatz einer Unterehmens-IT nur eine Teilmenge der Produkte verwendet werden müssen, die bei der Entwicklung z. B. zur Implementierung, benötigt werden. Dagegen muss die Software und Hardware auf der die Unternehmens-IT abläuft in beiden Fällen eingesetzt werden. Die drei Kostenposten werden im Folgenden untersucht.

5.2.5.2.1 Auswirkungen auf Personalkosten während Erstellung und Betrieb

Die Personalkosten gehen – bei konstanter Mitarbeiteranzahl – parallel mit dem Zeitbedarf zur Erstellung und für Aktivitäten während des Betriebs einer Unternehmens-IT zurück.[1163] Eine Softwarearchitektur wirkt auf unterschiedliche Weise auf diesen Zeitbedarf.

[1161] Vgl. Barry /Web Services & SOA/ S. 90.

[1162] Vgl. Sommerville /Software Engineering/ S. 520-521. Sommerville analysiert die Personalkosten einer Softwareentwicklung detaillierter und zählt zu Personalkosten auch Kosten der Bereitstellung des Arbeitsplatzes, für unterstützendes Personal, für Kommunikation und Netz, Sozialkosten und Ähnliches. In dieser Arbeit werden diese Kosten Sommerville folgend nicht explizit untersucht, sondern als Bestandteil der Personalkosten aufgefasst. Sommerville kommt in seinen Ausführungen zu dem Schluss, dass die Personalkosten in der Summe das Doppelte des Gehalts ausmachen. Die in dieser Arbeit ‚Produkt'-Kosten genannte Kostenart wird von Sommerville ‚Hardware- und Softwarekosten' genannt. Sommerville folgend werden in dieser Arbeit unter ‚Produkt'-Kosten sowohl Hardware- als auch Softwarekosten zusammengefasst.

[1163] Personalkosten können auch gesenkt werden, indem Gehälter gekürzt werden, Personal entlassen wird, Sozialleistungen gesenkt werden und Ähnliches. An dieser Stelle werden allerdings die Auswirkungen einer Softwarearchitektur auf Kosten untersucht und nicht, durch welche sonstigen Faktoren eine günstigere Softwareentwicklung erreicht werden kann. Aus diesem Grund fokussiert das Kapitel nur auf Zeitersparnisse, die durch den Einsatz einer SOA realisiert werden können.

Entwicklung einer SOA

In der Phase der Erstellung einer Software beeinflusst die Benutzbarkeit aus Entwicklersicht[1164] den Zeitbedarf zur Implementation einer Unternehmens-IT. Wenn eine Softwarearchitektur benutzbar (z. B. verständlich und betriebsfähig) ist, kann die zu entwickelnde Unternehmens-IT effizient erstellt werden. Es muss nicht bei jeder Implementation eines Teiles der Unternehmens-IT geprüft werden, ob bisherige Komponenten betriebsfähig sind oder ob die bestehenden Funktionen problemlos weiterverwendbar sind. Die Benutzbarkeit einer SOA wurde in Kapitel 3.2.4 sehr positiv bewertet, weshalb ein geringer Zeitbedarf erreicht werden kann.

Die Wiederverwendbarkeit[1165] einer Softwarearchitektur wirkt sich stark auf den Zeitbedarf zur Erstellung einer Unternehmens-IT aus. Die Wiederverwendbarkeit einer SOA wurde in Kapitel 3.2.4 sehr positiv bewertet. Teile einer Unternehmens-IT, die zur Erfüllung vieler Anforderungen verwendet werden können, müssen folglich nicht mehrfach implementiert werden. Dadurch wird Zeit gespart.

Zusammenfassend kann die sehr positive Bewertung[1166] der SOA in Bezug auf die Benutzbarkeit und Wiederverwendbarkeit eine Verringerung der Entwicklungszeit und somit auch eine Verringerung der Entwicklungskosten ergeben.[1167]

Demgegenüber stehen die Kosten für Analyse und Design. Zum einen entstehen diese, um Services im Funktionsumfang so voneinander abzugrenzen, damit diese in der Unternehmens-IT sinnvoll wiederverwendbar sind;[1168] zum anderen, um Schnittstellen gemäß den Gestaltungsrichtlinien einer SOA zu definieren.[1169] Weil in dieser Arbeit qualitative Bewertungen vorgenommen werden,[1170] kann für diesen Fall nicht eindeutig errechnet werden, ob eine SOA – unter Beachtung der aufgezeigten Wirkungszusam-

---

[1164] Vgl. Kapitel 3.1.3.4.2.

[1165] Vgl. Kapitel 3.1.3.7.2.

[1166] Vgl. Kapitel 3.2.4.

[1167] Vgl. hierzu auch Argumentationen von Stahlknecht, Hasenkamp /Einführung Wirtschaftsinformatik/ S. 323; Adam, McKendrick /Everything in Between/ S. 21.

[1168] Vgl. Kapitel 4.3.1.

[1169] Vgl. Kapitel 4.3.2.

[1170] Vgl. Kapitel 1.2.

menhänge – eine positive oder negative Auswirkung auf den Zeitbedarf zur Entwicklung einer Unternehmens-IT aufweist.[1171]

Betrieb einer SOA

Nach Irani und Baldwin birgt der langfristige Einsatz einer Unternehmens-IT die Unsicherheit der Produktivitätsreduzierung und des Verlustes von Wettbewerbsvorteilen.[1172] Die veralteten und dem Unternehmen nicht mehr angepassten Unternehmens-IT-Systeme sollten nicht weiterhin verwendet werden, wenn sich die Anforderungen an die Unternehmens-IT geändert haben, oder wenn eine bestmögliche Unterstützung des Unternehmens durch die Unternehmens-IT nicht mehr gegeben ist. Deswegen müssen während des Betriebs einer Unternehmens-IT Aktivitäten unternommen werden, um sie entsprechend aktueller Anforderungen betriebsfähig zu halten. Diese Aktivitäten werden in der Softwareentwicklung häufig „Wartung“[1173] genannt. Hierunter fallen die Reparatur von Softwarefehlern, die Anpassung an eine andere Betriebsumgebung und das Hinzufügen oder Ändern von Systemfunktionen.[1174] Um diesem Problem vorzubeugen, kann versucht werden, die Kosten notwendiger Änderungen an der Unternehmens-IT so gering wie möglich zu gestalten, damit die zu erwartenden Folgekosten keinen Grund zur Ablehnung der Modernisierung darstellen.

Alle Aktivitäten laufen auf eine Änderung der Software hinaus, weshalb eine hohe Wandlungsfähigkeit[1175] dazu führt, dass die entsprechend notwendigen Änderungen besser durchgeführt werden können als bei einer geringen Wandlungsfähigkeit.[1176] Eine

---

[1171] Eine solche Gegenüberstellung und eine anschließende Bewertung kann nur in der Praxis *für einen konkreten Fall* untersucht werden.

[1172] Vgl. zum folgenden Absatz Alshawi, Irani, Baldwin /Benchmarking/ S. 416-417.

[1173] Vgl. Sommerville /Software Engineering/ S. 612-621; Mellis /Projektmanagement/ S. 63-64. Mellis argumentiert, dass die Wartung weder eine eigenständige Aufgabe noch ein eigenständiges Entwicklungsprojekt darstellt, weil sie oft innerhalb gewöhnlicher Entwicklungsprojekte als neues Release einer Software umgesetzt werden. Diese Argumentation verträgt sich mit der Analyse der Kosten für Aktivitäten des Betriebs, weil diese unabhängig davon anfallen, wie diese Aktivitäten geplant und durchgeführt werden.

[1174] Vgl. Sommerville /Software Engineering/ S. 612-614.

[1175] Vgl. Kapitel 3.2.2.2.

[1176] Nach MacCormack, Verganti und Iansiti ist eine flexible Softwarearchitektur vor allem für Unternehmen in unsicherer und dynamischer Umgebung vorteilhaft. Vgl. MacCormack, Verganti, Iansiti /Products/ S. 135. Wie durch Trittmann u. a. empirisch gezeigt werden konnte, ist eine flexible Softwarearchitektur allerdings auch für eine Unternehmens-IT vorteilhaft, die einer geringen Anforderungsunsicherheit unterliegt und somit in einer eher sicheren und wenig dynamischen Umgebung agiert. Vgl. Trittmann u. a. /Sieg der Moderne/ S. 13-14.

SOA wurde in Kapitel 3.2.4 bezüglich der Wandlungsfähigkeit sehr positiv bewertet. Deshalb können anfallende Personalkosten während des Betriebs dementsprechend geringer ausfallen als bei der Verwendung einer weniger wandlungsfähigen Softwarearchitektur.

Demgegenüber steht jedoch ebenso wie bei der Erstellung einer Unternehmens-IT höherer Aufwand für die Entwicklung von Services und Schnittstellen.[1177] Dieser macht sich insbesondere bemerkbar, wenn bei der Abschaffung[1178] einer Schnittstelle alle Abhängigkeiten anderer Services berücksichtigt werden müssen. Wie oben dargestellt kann auch in diesem Fall nicht eindeutig ausgewertet werden, ob Kosteneinsparungen die Kosten der Investitionen ausgleichen. Hier wird aber vermutet, dass Kosteneinsparungen während des Betriebs einer SOA höher sind als die Kosten der zu tätigenden Investitionen, weil Aktivitäten der Entwicklung von Services und von Schnittstellen, z. B. durch die Erzeugung eines Domänenmodells[1179], nicht oder nur in sehr geringem Ausmaß durchgeführt werden müssen.[1180]

Struktur des Unternehmens

Eine Unternehmens-IT muss während der gesamten Lebensdauer eines Unternehmens betriebsfähig sein. Umfangreiche Änderungen innerhalb dieser Lebensdauer ergeben sich insbesondere durch geänderte strategische Ausrichtungen oder strategische Entscheidungen. Beispielsweise, wenn sich ein Unternehmen als Kostenführer positionieren will oder im Zuge strategischer Aktivitäten Fusionen bzw. Übernahmen anstrebt und durchführen muss. Weil strategische Änderungen für Unternehmen keine Seltenheit darstellen,[1181] ist eine Unternehmens-IT, die strategische Änderungen und verschiedene

---

[1177] Vgl. Kapitel 4.3.

[1178] Die Abschaffung einer Schnittstelle kann durchaus angestrebt werden, wenn diese beispielsweise nicht mehr in der Lage ist, korrekte Ergebnisse zu liefern. (Z. B. können gesetzliche Vorschriften in Kraft treten, die eine Erweiterung von Schnittstellen notwendig machen da existierende Schnittstellen den Vorgaben nicht genügen.) Durch die Fähigkeit einer SOA, Schnittstellen zu versionieren, kann i. d. R. davon ausgegangen werden, dass die Erstellung einer neuen Schnittstelle zu einem Service nicht notwendigerweise zu einer Abschaffung einer älteren Schnittstelle führt. Dadurch können Analyse- und Umbauaktivitäten zur Identifizierung von Abhängigkeiten reduziert werden.

[1179] Vgl. Kapitel 2.2.5.2.

[1180] Karch u. a. kommen zu einer vergleichbaren Einschätzung. Vgl. Karch u. a. /SAP/ S.24-25.

[1181] Vgl. Porter /Strategy/ S. 215. Porter stellt in seinem Buch Gründe für Strategieänderungen in verschiedenen Industrietypen dar. Er unterscheidet dabei ‚fragmented', ‚emerging', ‚declining' und ‚global industries'. (Vgl. Porter /Strategy/ Kapitel 9, 10, 12, und 13). Weil er jedem Unternehmen unterstellt, einen reifen Industrietyp (Industry maturity) anzustreben und auch dafür Gründe des

Strategien unterstützen kann, vorteilhaft. Sie verringert umfangreiche Anpassungen, wenn keine ausreichende Unterstützung vorhanden ist. Eine Softwarearchitektur als Gestaltungsgrundlage einer Unternehmens-IT hat großen Einfluss auf die Eigenschaften einer Unternehmens-IT. Weil SOA in Kapitel 3.2.4 sehr positiv bezüglich der Strategieunterstützung bewertet wurde, ergibt sich die Fähigkeit, die Kosten durch Strategieänderungen niedriger zu halten als bei Softwarearchitekturen, die eine geringere Strategieunterstützung aufweisen.

Eine Unternehmens-IT wird in der Regel viele Jahre betrieben.[1182] Eine Unternehmens-IT, die die zukünftigen Entwicklungen des Unternehmens unterstützt und sich mit dem Unternehmen entwickeln kann, kann über den Zeitraum des Betriebs bestehen. Eine nachhaltige Unternehmens-IT eröffnet das Potential, eine Unternehmens-IT vor ‚Überalterung' bewahren und die Notwendigkeit für Neuentwicklungen der gesamten Unternehmens-IT verhindern zu können. Eine solche Wirkung ist sehr langfristig zu betrachten und deswegen schwer bewertbar. Weil die Nachhaltigkeit einer SOA jedoch sehr positiv bewertet wurde,[1183] kann von einer positiven Auswirkung auf die Kosten des Betriebs einer Unternehmens-IT ausgegangen werden.

5.2.5.2.2 Auswirkungen auf Kosten der Produkte

Die Betrachtung von technischen Ausprägungen einer SOA und folglich auch von konkreten Produkten wird in dieser Arbeit ausgeklammert.[1184] Eine Auswirkung kann jedoch identifiziert werden, die unabhängig von der Wahl einer Technologie oder spezifischen Produkten existiert.

Eine Integration aller Teile einer Unternehmens-IT in eine integrierte Unternehmens-IT ermöglicht die Konsolidierung aller eingesetzter Komplementärprodute.[1185] Durch die

---

Strategiewandels nennt (vgl. Porter /Strategy/ Kapitel 11), kann davon ausgegangen werden, dass jedes Unternehmen, für dessen Unternehmens-IT der Einsatz einer SOA sinnvoll erscheint, Strategieänderungen ausgesetzt ist.

[1182] Vgl. Sommerville /Software Engineering/ S. 589; Kirchmer /Einführung/ S. 43.

[1183] Vgl. Kapitel 3.2.3.2.

[1184] Vgl. Kapitel 1.3.3; Konkrete Technologie oder ein Produkt zum Einsatz für eine SOA können ebenso mit dem Qualitätsmodell aus Kapitel 3.1 bewertet werden. Einige Autoren versuchen sich in der Bewertung von Produkten auf allgemeiner Ebene, kommen allerdings zu ebenso allgemeinen Ergebnissen. Beispielsweise Karch, der vorschlägt, nur verbreitete Technologien zu verwenden. Vgl. Karch u. a. /SAP/ S.24-25.

[1185] Vgl. zum folgenden Absatz Karch u. a. /SAP/ S.24-25, S. 37.

Möglichkeit, den Produktmix zum Betrieb einer Unternehmens-IT zu reduzieren, können sich Einsparpotentiale bezüglich Softwarekosten (Lizenz- & Produktkosten) als auch Wartungskosten ergeben. Darüber hinaus kann sich auch der Zeitbedarf für das Management und die Verwaltung der Produkte verringern und dadurch können Personalkosten gespart werden.

Andererseits bedingt der Einsatz einer SOA auch den Einsatz spezieller Produkte, z. B. einer Kommunikationsinfrastruktur mit ihren Bestandteilen.[1186] Solche Produkte müssen angeschafft, eingeführt und verwaltet werden und verursachen folglich Kosten. Da diese Produkte jedoch einheitlich für die gesamte Unternehmens-IT Anwendung finden und gleichzeitig eine Konsolidierung übriger Softwareprodukte angestrebt wird, wird davon ausgegangen, dass insgesamt und langfristig eine Verringerung der Kosten erreicht werden kann.

5.2.5.2.3 Auswirkungen auf Personalkosten der Verwendung

Personalkosten der Verwendung einer Unternehmens-IT fallen auf der Seite der Fachabteilungen[1187] eines Unternehmens an. Sie betreffen Harmonisierungsaspekte und vermeidbare Personalkosten.

Harmonisierung des Unternehmens und der IT

Eine ‚Harmonisierung zwischen IT und Unternehmen'[1188] hat zum Ziel, dass einerseits die Anforderungen, Einschränkungen und Fähigkeiten der IT durch das Unternehmen bzw. durch die Fachabteilungen erkannt und genutzt werden; andererseits, dass die Anforderungen, Einschränkungen und Fähigkeiten des Unternehmens durch die IT auch erkannt und genutzt werden. Die Harmonisierung soll erreichen, dass sowohl die Strukturen als auch das Verhalten der Unternehmens-IT und die des Unternehmens nicht gegensätzlich ausgeprägt sind und sich gegenseitig behindern. Mit der Harmonisierung wird oft der Vorteil verbunden,[1189] dass Geschäftskomplexität in Komponenten gekap-

---

[1186] Vgl. Kapitel 2.2.4.3.

[1187] Unter der Fachabteilung wird hier der Nutzer(-kreis) verstanden, der die Unternehmens-IT einsetzt, um die ihm gestellten betrieblichen Aufgaben erfüllen zu können.

[1188] Vgl. Gronau /Wandlungsfähige Informationssystemarchitekturen/ S. 216.

[1189] Vgl. zum folgenden Absatz Banke, Krafzig, Slama /Enterprise SOA/ S. 79-80.

selt wird, wodurch sich die Präsentationsebene auf Präsentation und Ablaufkontrolle beschränken kann. Eine SOA bietet das Potential, dass eine Harmonisierung erreicht werden kann, weil sich zum einen Services an Geschäftsprozessen orientieren sollen und deswegen die Geschäftsprozesslandschaft eines Unternehmens durch die Unternehmens-IT abgebildet und von den Entwicklern der Unternehmens-IT vollständig verstanden werden muss; zum anderen, weil das Qualitätsmerkmal der Nachhaltigkeit durch die sehr positive Bewertung[1190] einer SOA attestiert, Strukturen und Vorgänge gut abbilden zu können. Es kann zwar nicht sichergestellt werden, dass das Unternehmen bzw. die Fachabteilungen die Anforderungen, Einschränkungen und Fähigkeiten der IT vollständig erfassen und einsetzen können. Oft bedeutet die Bildung eines umfassenden Abbildes der Strukturen und der Vorgänge in einer Unternehmens-IT ein Aufwandsanstieg in Form von Personalkosten.

Einfluss auf Fachabteilungen

Vermeidbare Personalkosten entstehen in Fachabteilungen, wenn eine Unternehmens-IT den Anforderungen nicht genügt und Aufgaben umständlicher und infolgedessen zeitaufwändiger erledigt werden müssen als wenn Anforderungen genau erfüllt würden. Diese Ineffizienzen sind oft die Folge von unterlassenen aber notwendigen Änderungen, weil diese zu aufwändig sind. Die Wandlungsfähigkeit einer SOA wurde sehr positiv bewertet[1191] und hilft dadurch Personalkosten zu vermeiden.

Personalkosten lassen sich auch in der Verwendung einer Unternehmens-IT beeinflussen, wenn sie verlässliche Eigenschaften aufweist. Die Verlässlichkeit einer SOA wurde in Kapitel 3.2.4 sehr positiv bewertet. Folglich werden Mitarbeiter nicht davon abgehalten, ihre Arbeit zu erledigen, weil die Unternehmens-IT oder Teile von dieser ausgefallen sind.

Personalkosten lassen sich auch durch das Antwortzeitverhalten einer Unternehmens-IT beeinflussen. Wenn Mitarbeiter auf Reaktionen seitens der Unternehmens-IT zur Abarbeitung von Aufgaben lange warten müssen, ohne dass in der Zwischenzeit sinnvoll an anderen Aufgaben gearbeitet werden kann, ergibt sich eine unproduktive Zeit bei den

---

[1190] Vgl. Kapitel 3.2.4.

[1191] Vgl. Kapitel 3.2.4.

Mitarbeitern. Eine SOA wurde bezüglich der Effizienz (und des Antwortzeitverhaltens) negativ bewertet[1192] und wirkt sich demgemäß auch negativ auf die Personalkosten aus.

Als Letztes lassen sich Personalkosten einsparen, wenn automatisierbare Aufgaben durch die Unternehmens-IT übernommen werden können. Dies betrifft auch die Kommunikation mit Externen. Geben Kunden ihre Daten z. B. selber ein, oder informieren sich Lieferanten selbstständig über die Bedarfsmenge eines Produktes, können Mitarbeiter von Aufgaben der Informationsweitergabe entlastet werden. Eine solche Möglichkeit ergibt sich, wenn die Unternehmens-IT eine Integrationsreichweite über Unternehmensgrenzen hinweg aufweist. Im Falle einer SOA geht die Integrationsreichweite über Unternehmensgrenzen hinaus und die Integriertheit wurde sehr positiv bewertet,[1193] weshalb sich bzgl. der Personalkosten Vorteile durch SOA ergeben.

5.2.6 Zusammenfassung

Zusammenfassend ergibt sich das in Tab. 5-1 dargestellte Ergebnis. Es ist ersichtlich, dass der Einsatz einer SOA überwiegend positive Wirkungen auf die Wirtschaftlichkeit zur Folge hat. Einige der Wirtschaftlichkeitskriterien konnten jedoch auch auf qualitativer Ebene nicht eindeutig bewertet werden.[1194] Auch für diese Bewertung muss hervorgehoben werden, dass sie keine spezifische Situation betrachtet und bei einer Überprüfung einer spezifischen Situation ggf. unterschiedliche Ergebnisse eintreten können.

Wirtschaftlichkeitskriterium		**Wirkung**[1195]
Kunden-zufriedenheit	Lieferzeiten	+
	Kooperative Auftragsabwicklung	+

[1192] Vgl. Kapitel 3.2.4.

[1193] Vgl. Kapitel 3.2.4.

[1194] Für eine eindeutige Bewertung würde die Notwendigkeit bestehen, das entsprechende Wirtschaftlichkeitskriterium weiter zu verfeinern. In der Argumentation des Qualitätsattributes ist dies durchgeführt worden, würde in der Zusammenfassung allerdings zu einem unverhältnismäßigen Eindruck der Bewertung führen. Aus diesem Grund wird hier dargestellt, dass eine wirtschaftliche Bewertung bezüglich verschiedener Argumentationen unterschiedlich ausgefallen ist.

[1195] Legende: [+] Positiv, [0] Neutral, [–] Negativ, [+, –] für dieses Wirtschaftlichkeitskriterium kann keine eindeutige Wirkung festgestellt werden, bzgl. der verschiedenen Auswirkungen vgl. die jeweilige Argumentation im Text.

Wirtschaftlichkeitskriterium		Wirkung[1195]
	Innovative Produkte	+
	Qualität der Produkte	+
Mitarbeiter-zufriedenheit	Daten und Prozesse	+
	Qualität der Unternehmens-IT	+
	Arbeitsprozess	+, –
	Komplexität in der Entwicklung	+
	Softwareentwicklungsprozess	+
Wettbewerbs-situation	Automatisierung	+
	Vorhandene Wettbewerbsvorteile	+
	Anpassung der Wertschöpfungskette	+
	Wettbewerbsbedingte Notwendigkeit	+
	Verarbeitungsgeschwindigkeit	–
	Anpassung des Umfeldes	+
Beziehung zu Geschäftspartnern	Operative Zusammenarbeit	+
	Gestaltung der Geschäftsprozesse	+, –
Interne Wirkungen	Entwicklung einer SOA	+, –
	Betrieb einer SOA	+
	Struktur des Unternehmens	+
	Kosten der Produkte	0 (+)[1196]
	Harmonisierung des Unternehmens und der IT	+, –
	Einfluss auf Fachabteilungen	+, –

Tab. 5-1: Zusammenfassung des Wirtschaftlichkeitspotentials

[1196] Siehe Text, Kapitel 5.2.5.2.2.

5.3 Realisierung des Wirtschaftlichkeitspotentials

Wie Brynjolfsson und Yang herausgearbeitet haben lässt sich der Nutzen der IT für die Gesamtwirtschaft nicht bewerten.[1197] Viele andere Faktoren haben Einfluss auf die Wirtschaftlichkeit und IT ist nur einer von ihnen. Wie mehrfach dargestellt, können die verschiedenen Potentiale einer SOA – das Nutzen-, Kosten und Wirtschaftlichkeitspotential[1198] – nicht einheitlich für alle Unternehmen bewertet werden. Daraus lässt sich schlussfolgern, dass der Einsatz einer SOA für unterschiedliche Firmen verschiedene Ausprägungen bezüglich der Potentiale ausweist.

Die daraus resultierende Frage ist: Wie vorteilhaft ist der Einsatz einer SOA für ein Unternehmen? Eine solche Fallstudie soll an dieser Stelle nicht durchgeführt werden.[1199] Jedoch kann kurz beispielhaft aufgezeigt werden, welche Eigenschaften eines Unternehmens bzw. eines Umfelds eines Unternehmens dazu führen, dass der Einsatz einer SOA entsprechend der Analyse dieser Arbeit bedeutend lohnend ist.

Unternehmen unterscheiden sich nicht in grundlegenden Geschäftsfunktionen.[1200] Die hauptsächlichen Geschäftsfunktionen, die fast jedes Unternehmen aufweist, sind Produktionsplanung und -steuerung, Controlling, Materialwirtschaft, Vertrieb, Finanz- und Rechnungswesen sowie Personalwirtschaft.[1201] Darüber hinaus existieren in jedem Unternehmen Merkmale der Spezialisierung und Hierarchie. Jedes Unternehmen agiert in einem Ökosystem. Zu einem Ökosystem eines Unternehmens gehören Lieferanten, Wettbewerber, Regulatoren, Teilhaber und Kunden. Weiterhin gehören dazu Eigenschaften des Wirtschaftssystems, der Politik, Internationalisierungs- bzw. Globalisierungsausprägungen, Technologien und Ausprägungen der Wissenschaft. Anhand dieser Kriterien lassen sich Segmentierungen von Unternehmen vornehmen. Eine weit verbreitete Dimension stellt z. B. die Branche dar, in der ein Unternehmen aktiv ist. Die Dimension einer Branche definiert sich dabei über das Merkmal der Spezialisierung und über Teile des Ökosystems. Ausprägungen der Automobilbranche z. B. sind die Spezia-

---

[1197] Vgl. Brynjolfsson, Yang /Productivity/ S. 187-192, 207.

[1198] Da das Wirtschaftlichkeitspotential auf dem Nutzen- und Kostenpotential aufbaut, wird im Folgenden nur noch vom Wirtschaftlichkeitspotential gesprochen.

[1199] Dies würde den Rahmen dieser Arbeit um ein Vielfaches überschreiten, stellt jedoch einen Punkt für weitere Forschung bzw. für den Einsatz in der Praxis dar. Vgl. Kapitel 1.

[1200] Vgl. zum folgenden Absatz Laudon, Laudon /Business/ S. 35-40.

[1201] Vgl. Hansen, Neumann /Wirtschaftsinformatik/ S. 529-532.

lisierung auf den Bau von Automobilen und verbundenen Dienstleistungen sowie das Agieren in einem globalen Markt. Eine Bewertung des Wirtschaftlichkeitspotentials aller Branchen würde den Umfang dieser Arbeit wesentlich überschreiten. Ebenfalls kann angenommen werden, dass die Bedeutung einer SOA für einzelne Unternehmen nur fallabhängig vorgenommen werden kann, weil sich Unternehmen einer Branche stark unterscheiden können.[1202] Ähnliche Argumente existieren für alle Kriterien, anhand deren sich Unternehmen segmentieren lassen würden. Die folgende Ausarbeitung ist ein Beispiel für einen vorteilhaften Einsatz einer SOA.[1203]

Potentiell lohnenswerter Einsatz einer SOA

Der Einsatz einer SOA wird sich für die Unternehmen lohnen, die eine Unternehmens-IT betreiben und in einem Umfeld agieren, in dem

1. von Kunden und Lieferanten erwartet wird, dass Geschäftsprozesse und Geschäftsabschlüsse in Echtzeit[1204], zumindest aber am Handelstag, vollständig abgewickelt werden können,
2. zu vielen Geschäftspartnern aktive Geschäftsbeziehungen existieren und diese global agieren,
3. Produkte zum großen Teil darauf angewiesen sind, von IT-Systemen abgebildet und verwaltet zu werden,
4. Produkte in kurzen Produktzyklen oder in Produktvariationen in kurzen Abständen auf den Markt kommen und
5. sich die Wettbewerberstruktur regelmäßig ändert.

[1202] Beispielsweise wird ein Zulieferer eines Automobilherstellers andere Anforderungen an eine Unternehmens-IT aufstellen als ein Fahrzeugfabrikant, obwohl beide in der gleichen Branche tätig sind.

[1203] Prinzipiell sei darauf hingewiesen, dass eine Einführung einer SOA nur für Unternehmen in Frage kommt, die eine umfangreiche Unternehmens-IT einsetzten (wollen). Unternehmen führen IT-Systeme ein, um zwei Problemen begegnen zu können: Zum einen sollen interne Gruppen, die die Produkte und Services erstellen, effizient gemanagt werden; zum anderen soll mit dem Unternehmens-Umfeld, z. B. mit Kunden und Lieferanten, effizient interagiert werden. Dies lässt sich ab einem bestimmten Umfang nur noch mit Hilfe einer Unternehmens-IT bewerkstelligen.

[1204] Ein in der anglistischen Literatur ‚time to market' genannter Zeitfaktor. Vgl. hierzu Bass u. a. /Software architecture/ S. 95.

Beispiele für Unternehmen, auf die diese Kriterien zutreffen können, sind Finanzinstitute (insb. Versicherungs- und Wertpapierhandel), Unternehmen des Energiemarktes und Unternehmen der Telekommunikationsbranche.

Die Verwendung einer SOA unter diesen Rahmenbedingungen als Eigenschaften eines Ökosystems eines Unternehmens führt zu Anforderungen an eine Unternehmens-IT, sodass ein positives Wirtschaftlichkeitspotential aus der Verwendung einer SOA entsteht.

Erklären lässt sich dies folgendermaßen: Das oben skizzierte Ökosystem stellt hohe Anforderungen an die Wandlungsfähigkeit einer Unternehmens-IT. Produkte können kurzzeitig Änderungen unterliegen, der Wettbewerbermarkt kann fluktuieren und Geschäftsbeziehungen zu vielen Geschäftspartnern und Kunden können bestehen oder aufgebaut werden.[1205] Die Anforderungen an die Unternehmens-IT sind hoch, weil diese Änderungen kontinuierlich und in umfangreichem Ausmaß auftreten. In der Versicherungsbranche z. B. entscheidet der Zeitbedarf, den eine Unternehmens-IT benötigt, um die entsprechenden Anforderungen des Produktes abbilden zu können, wann neue Produkte auf den Markt kommen.[1206] Die umfangreichsten Anforderungen ergeben sich jedoch, wenn ein Wettbewerber übernommen wird; in etwas verringertem Umfang, wenn eine Kooperation mit einem Wettbewerber eingegangen wird, um gemeinsam gegenüber anderen Wettbewerbern bestehen zu können. Beide Unternehmens-IT-Systeme müssen aufeinander abgestimmt werden können. Alle diese Faktoren bedingen eine Änderung der Unternehmens-IT. Eine wandlungsfähige Unternehmens-IT hat für diese Situation gegenüber weniger wandlungsfähigen Systemen einen großen Vorteil, weil Änderungen wettbewerbsrelevant sind und schnell durchgeführt werden können.

In dem skizziertem Umfeld ist eine intensive Kommunikation und Interaktion mit Geschäftspartnern zur Bearbeitung von Geschäftsprozessen notwendig. Darüber hinaus werden die Anforderungen gestellt, dass diese Kommunikation und Interaktion in sehr kurzer Zeit und in großer Menge durchgeführt werden kann. Um dies erfüllen zu können, müssen die Geschäftsprozesse, die über Unternehmensgrenzen hinaus durchgeführt werden oder die Informationen von externen Quellen benötigen, vollautomatisch durch-

---

[1205] Vgl. Berensmann /Postbank/ 65.

[1206] Vgl. Moormann /Bankinformatik/ S. 6-7.

geführt werden können.[1207] Wenn andere Unternehmens-IT-Systeme am Geschäftsprozess beteiligt sind, z. B. Börsen und Wertpapierhändler, ist ein vollautomatisch durchgeführter Geschäftsprozesses möglich. Ist der Kunde z. B. ein Fondsmanager, kann sogar eine Integration der Unternehmens-IT des Kunden mit dem des Wertpapierhändlers angestrebt werden. Weil die Integriertheit sehr positiv bewertet wurde, kann eine SOA hier nicht nur eine Verbesserung der operativen Zusammenarbeit erreichen, sondern zu einer enormen Effizienzsteigerung des gesamten Geschäftsprozesses – vom Kunden bis zum Lieferanten – führen. Eine Verbesserung der Beziehungen zu Geschäftspartnern wird erreicht, und darüber hinaus kann die Kunden- und Mitarbeiterzufriedenheit gesteigert werden. Ferner können Wettbewerbsvorteile und Kostenersparnisse realisiert werden. Eine detaillierte Betrachtung müsste selbstverständlich situationsspezifisch durchgeführt werden.

Nachteilig wirkt sich eine SOA aufgrund dieser Anforderungen aus, weil in Kapitel 3.2.4 die Effizienz negativ und die Portabilität sehr negativ bewertet wurden. Zusätzliche Anstrengungen müssen unternommen werden, um erstens ein notwendiges Effizienzlevel, z. B. bei Antwortzeitverhalten und Ressourcennutzung, erreichen zu können: zweitens aber auch, um eine Integration für die hohe Anzahl an Kunden und vor allem Geschäftspartnern ermöglichen zu können. Diese Aktivitäten wirken sich nicht negativ auf die Gesamtbewertung aus, da sich der Nutzen potentiell höher einstufen lässt, weil eine hohe Anzahl an Kunden und Geschäftspartnern integrierbar ist.

[1207] Kirchmer und Scheer stellen dar, dass Kriterien, die im Beispiel zu finden sind, zu Änderungen von Geschäftsprozessen führen. Dies sind vor allem: neue Kunden oder Kunden, deren Bedürfnisse sich ändern, Zulieferer oder andere Marktteilnehmer, neue oder veränderte Marktangebote und Fusionen bzw. Übernahmen. Vgl. Kirchmer, Scheer /Change Management/ S. 3-5.

6. Wirtschaftlichkeitsbetrachtung einer SOA in der Praxis

In dieser Arbeit wurde ein Qualitätsmodell zur Bewertung von Softwarearchitekturen aufgestellt und anhand dessen das Nutzenpotential einer SOA bewertet. Anschließend wurde das Kostenpotential dargestellt und schließlich eine Wirtschaftlichkeitsanalyse durchgeführt. Das Vorgehen beruht sehr stark auf dem Qualitätsmodell zur Nutzenbeurteilung einer Softwarearchitektur. Um aufzuzeigen, dass das vorgestellte Bewertungsverfahren einen signifikanten Beitrag zum Erkenntnisstand liefert und hilft, die Anforderungen der Bewertung einer SOA besser zu verstehen, wurde eine Interviewreihe mit fünf Praktikern durchgeführt. Durch den Fokus auf Softwarearchitekten, die Bewertungen von mindestens einer Softwarearchitektur vorgenommen haben und möglichst speziell SOA zu bewerten hatten, konnte keine große Anzahl an Interviewpartnern erreicht werden. Schließlich konnte eine Auflage von fünf Interviews erreicht werden. Das erhobene Datenmaterial zeigt sich mehr als ausreichend und sehr geeignet für die Erhebung.

Drei der Interviewpartner[1208] befassen sich mit der Bewertung von SOA, die anderen bewerten Softwarearchitekturen im Allgemeinen.[1209] Mit der Begründung, dass SOA aktuell ein Schlagwort ist, das in der Softwarebranche intensiv diskutiert wird, haben alle Interviewpartner nur unter der Bedingung einem Interview zugestimmt, dass alle erhobenen Daten mindestens anonymisiert sind.[1210] Weil einige Interviewpartner darauf bestanden, dass die erhobenen Daten nicht nachvollziehbar sein sollen, wird bei der Darstellung der Daten im Folgenden keine Zuordnung zu Interviewpartnern vorgenommen. Stattdessen werden alle Daten einer Frage gemeinsam vorgestellt, und die Reihenfolge der Interviewpartner wurde von Frage zu Frage zufällig geändert. Die Interviews wurden im September und Oktober 2005 durchgeführt und dauerten zwischen 90 und 110 Minuten. In zwei Fällen war ein telefonisches Nachfassen notwendig.

[1208] Um eine bessere Lesbarkeit zu erreichen wird in dieser Arbeit der Begriff ‚Interviewpartner' unabhängig vom Geschlecht des Interviewpartners nur in der männlichen Form verwendet.

[1209] Die Bewertung einer SOA stellt natürlich auch die Bewertung einer Softwarearchitektur dar. Somit sind die Interviewpartner, die eine SOA bewertet haben, auf die Bewertung von SOA spezialisiert, und es wird erwartet, dass diese ein Bewertungsmodell für SOA aufgestellt haben, das zu besseren Ergebnissen führt als die der anderen Interviewpartner. Dabei wird angenommen, dass bessere Ergebnisse erreicht werden, wenn ein Bewertungsmodell dem Bewertungsmodell dieser Arbeit ähnlich ist.

[1210] Dies ließ sich vor allem darauf zurückführen, dass vier der Interviewpartner aus der IT-Beratungsbranche stammen und teilweise insbesondere Bewertungen einer SOA durchführen.

Aus den Interviews ergibt sich, dass die Praxis nicht so weit in der Entwicklung ist, alle Aspekte zur Bewertung einer SOA zu beachten. Die Arbeit liefert einen signifikanten Beitrag zum Erkenntnisstand und hilft, die Problematik der Bewertung einer SOA besser zu verstehen.

Die Form eines Interviews wurde gewählt, weil das Risiko besteht, dass Ergebnisse verfälscht werden. Eine Alternative zum Interview sind z. B. Fragebögen. Bei Fragebögen besteht jedoch die Unsicherheit, dass sich Teilnehmer bei offener Fragestellung nicht genügend Zeit zur Beantwortung nehmen und deswegen zu wenige Daten erhoben werden. Zudem kann es zu Missverständnissen kommen. Diese Unsicherheiten werden bei einem Interview umgangen bzw. können durch direkte Rückfragen vermieden werden.

Im Folgenden Kapitel 6.1 wird das erhobene Datenmaterial dargestellt. Im Kapitel 6.2 wird eine Analyse vorgenommen und das Ergebnis diskutiert.

6.1 Datenmaterial

Im Folgenden wird das erhobene Datenmaterial dargestellt. Dazu wird in Kapitel 6.1.1 zunächst der Interviewleitfaden dargestellt. In Kapitel 6.1.2 werden die Ergebnisse der Datenerhebung zusammenfassend dargestellt.[1211]

6.1.1 Interviewleitfaden

Der Interviewleitfaden gliedert sich in drei Abschnitte (Fragenkomplexe): Fragen zum Bewertungsvorgehen, Erhebung des Erfolgs der vorgenommenen Bewertung (sofern bewertbar), Auskünfte über den Interviewpartner.

Insgesamt wurden sechs offene Fragen zum Bewertungsvorgehen gestellt. Die offene Fragestellung ermöglichte es dem Interviewpartner, auf einen großen Teil der Fragen zu antworten, ohne dass diese explizit gestellt werden mussten. Die Fragen sind: Wie haben Sie die Wirtschaftlichkeit von SOA-Projekten bewertet? Welche Qualitätsattribute haben Sie betrachtet? Haben Sie zwischen technisch-fachlichen und geschäftlichen Qualitätsattributen unterschieden? In welchen Bereichen haben Sie Kosten identifiziert?

[1211] Werden Interviewpartner im Folgenden zitiert, wird dies durch doppelte Hochkommata, jedoch ohne Referenz angegeben, um der Anforderung der Anonymität gerecht werden zu können.

Wie viel Aufwand haben Sie in die Bewertung investiert? Ist der betriebene Aufwand zur Bewertung Ihrer Meinung nach angemessen?

Die Erhebung des Erfolgs konnte auf freiwilliger Basis ergänzt werden. Hier sollte die Eignung der dargestellten Bewertungsmethode zur Architekturbewertung bzw. zur Bewertung einer SOA festgestellt werden. Die Interviewpartner sollten sich möglichst auf ein Projekt beziehen, welches eine erfolgreiche Bewertung (nicht unbedingt eine positive Bewertung) hatte. Es wurden die beiden Fragen gestellt, was das Ergebnis der Bewertung war (positiv oder negativ), und ob das Ergebnis schon verifiziert werden konnte. Mittels dieser Daten sollte erstens eine Korrelation bestimmter Qualitätsattribute auf bestimmte Ergebnisse der Bewertung vorgenommen werden; zweitens wurde über die Verifizierung angestrebt, einen ersten Eindruck darüber zu erhalten, wie gut die angewendete Bewertungsmethode war.

Der dritte Fragenblock bezog sich auf den Interviewpartner. Hier wurden drei Fragen gestellt: Wie viele SOA-Projekte (alternativ: Softwarearchitektur-Projekte) haben Sie bisher bewertet? Welchen Umfang (in Personentagen) hatten diese Projekte? Wann fanden diese Bewertungen statt? Diese Daten dienen der Absicherung, dass eine gute Datenqualität erreicht werden konnte, weil qualifizierte Interviewpartner ausgewählt wurden. Die Beantwortung dieser Fragen war freiwillig.

Während des gesamten Interviews stand es den Interviewpartnern frei, alles darzulegen, was ihnen für ihre Bewertung und ihr Bewertungsvorgehen wichtig erschien. Einige Interviewpartner haben Informationen gegeben, die für diese Arbeit nicht weiter relevant sind, z. B. die Bewertung einer Technologie. Die Informationen zur Bewertung der Technologie werden in dieser Arbeit deswegen nicht dargestellt.[1212]

6.1.2 Erhobenes Datenmaterial

Im Folgenden werden die erhobenen Daten nach den Fragenkomplexen gegliedert dargestellt.

[1212] Dem Autor ist bewusst, dass sich Qualitätsattribute zur Bewertung einer Technik durchaus auf die Bewertung einer SOA übertragen lassen (z. B. die Benutzbarkeit aus Entwicklersicht). Wurden diese Qualitätsattribute vom Interviewpartner im Zusammenhang mit einer spezifischen Technologie oder einem spezifischen Produkt erwähnt, wurde das Qualitätsattribut als beachtet interpretiert.

6.1.2.1 Fragenkomplex 1: Fragen zum Bewertungsvorgehen

Frage 1a: Wie haben Sie die Wirtschaftlichkeit von SOA- (bzw. Softwarearchitektur)-Projekten bewertet?

Auf diese Frage haben drei Interviewpartner eine fast übereinstimmende Antwort gegeben: „Mittels der Bewertung von Nutzen und Kosten“. Einer dieser Interviewpartner fügte dem noch „Risiken“ hinzu, zwei beschrieben diese Kosten-Nutzen-(Risiken-)Betrachtung als „Business-Case“-Bewertung. Unter einer „Business Case“-Bewertung verstanden sie die Fokussierung auf Teilaspekte einer Softwarearchitektur, z. B. Schnittstellen, und identifizierten für diese Fokussierung Kosten, Nutzen und (einer der Interviewpartner) Risiken.

Einer der Interviewten bewertete acht „Business Cases“. Er fokussierte die Bewertung dabei auf die Teilaspekte Fehlerreduzierung, Wartungsaufwand, Restrukturierung der Unternehmens-IT, Releasefähigkeit, Schnittstellen, Testen, Softwarekonsolidierung und „Intangibles“. Im Business Case „Intangibles“ wurden Kommunikationsaspekte, Spezifikationsaspekte, Softwareflexibilität und Softwareerweiterbarkeit betrachtet. Der Interviewpartner erwähnte noch, dass die „Wirtschaftlichkeit aus Kundensicht“ bewertet werden müsse, dies wurde jedoch aus Ressourcenmangel nicht durchgeführt.

Ein anderer Interviewpartner führte „Business Cases“ anhand der Kriterien Entwicklung von Fehlerraten, Aufwand im Betrieb, Aufwand der Entwicklung (Zeitbedarf), Frequenz von Releases und Anpassbarkeit an Geschäftspartner durch.

Einer der drei Interviewpartner gab an, dass insbesondere der Kostenaspekt – Einsparungen als auch Mehrausgaben – intensiv untersucht wurde. Wesentliche monetäre Größen, die bewertet wurden, waren variable Kosten, Fixkosten und Erlöse. Variable Kosten einer SOA sind zum einen „die gleichen wie bei jedem anderen Projekt: z. B. Durchlaufzeiten, Personalkosten und Dauer der Ressourcenbindung.“ Zum anderen existieren auch SOA-spezifische variable Kosten, z. B. die „Erstellung eines Businessmodells[1213].“ Fixkosten sind Kosten der Anschaffung von Komplementärprodukten. Erlöse wurden je Geschäftsprozess bewertet. Hierbei wurden drei Parteien betrachtet:

[1213] Das Businessmodell (später vom Interviewpartner als „Domain Modell“ bezeichnet) stellt die Grundlage des „Business Design“ dar. Die Beachtung des „Business Design“ erst sorgt für Veränderungen gegenüber nicht-SOA-Projekten. Das „Business Design“ ist der Prozess der Definition von Services.

Kunden, Lieferanten und interne Prozesse. Insbesondere wurde bewertet, ob eine Umsatzsteigerung erreicht werde kann, indem Services für Externe angeboten werden. Das Businessmodell erlaubt es, Servicekandidaten anhand ihrer Wertschöpfung zu typisieren. Deswegen konnte eine A-B-C-Analyse[1214] durchgeführt werden mit dem Ziel, (zunächst) nur die Typ A–Services zu realisieren.

Ein anderer Interviewpartner stellt ein Bewertungsvorgehen vor, in dem zunächst festgelegt wird, „was" bewertet werden soll und dass – darauf aufbauend – eine „Zusatzkosten-Bewertung" stattfindet. Das „Was" stellt im Fall der Bewertung einer SOA die Anbindung und Erstellung von Services, Kosten auf Fachseite (z. B. Personalaufwand) und Kosten der Kommunikation mit der Fachseite. Dabei wurde zweistufig vorgegangen: Zunächst wurde bestimmt, ob und in welcher Höhe ein „Return on Investment"[1215] (ROI) realisiert werden kann. Anschließend, wie ein erweitertes Kostenmodell für das konkrete Projekt aussieht. Das erweiterte Kostenmodell stellt das im weiteren Verlauf des Interviews vorgestellte Qualitätsmodell des Interviewpartners dar. Die ‚Kosten der Fachseite' wurden durch den Interviewpartner als schwer erhebbar und deswegen nicht verwendet dargestellt. Der Interviewpartner stellte fest, dass das Vorgehen nur für konkrete Fälle anwendbar ist. In die ROI-Bewertung flossen Personalaufwand und Kosten für Komplementärprodukte (insb. Middleware) ein. Dem Interviewpartner nach sind weitere Daten für eine ROI-Bewertung schwierig zu beschaffen und stellen eine „Hintergrundbetrachtung"[1216] dar. Ein grundlegendes Problem einer ROI-Bewertung sei die „Schönrechnungsgefahr"[1217].

Der fünfte Interviewpartner stellte fest, dass er die „Wirtschaftlichkeit fast nicht" bewertet hat. Stattdessen hat er „Architekturbewertungen für Integrationsarchitekturen" vorgenommen. Ziel der Bewertungen war, eine Evaluation der Architektur bezüglich der Eignung für (weitere) Projekte durchzuführen. Explizit bewertet wurde hauptsächlich die Zukunftssicherheit einer Softwarearchitektur für konkrete Projekte. Das vom Inter-

---

[1214] Vgl. Günther, Tempelmeier /Produktion/ S. 181-183.

[1215] Vgl. zum ‚Return on Investment' mit Bezug zur Software Solingen /ROI/ oder die ROI-Spezialausgabe des Cutter IT Journals (Cutter IT Journal „Analyzing IT ROI: Can We Prove the Value?" Vol. 18, Nr. 8, 2004).

[1216] Unter diesem Begriff versteht der Interviewpartner schwer bewertbare Kriterien. In dieser Arbeit wurden diese ‚intangibel' genannt.

viewpartner vorgestellte Qualitätsmodell umfasst Qualitätsattribute zur Bewertung von Integrationsarchitekturen, ohne dass daraus direkt eine Wirtschaftlichkeit abgeleitet werden kann.

Drei Interviewpartner hoben hervor, dass es zur Kontrolle einer Bewertung wichtig ist, Kriterien zu bewerten, die später auch gemessen und anhand derer ggf. Korrekturmaßnahmen ergriffen werden können, sog. ‚Key Process Indicators' (KPI). Einer der Interviewpartner schlägt vor, Messpunkte für Wartungsaufwand, Anzahl realisierter Funktionen und Granularität von Services alle sechs Monate zu überprüfen. Ein anderer Interviewpartner will Durchlaufzeiten, Personalaufwand und Ressourcenbindung nach zwei bis drei Jahren kontrollieren. Einig waren sich die drei Interviewpartner, dass Kontrollpunkte und Kriterien festgelegt werden sollten.

Zwei Interviewpartner haben bei der Bewertung einer SOA auch Technologien bewertet. Diese Interviewpartner haben den Reifegrad und die Beherrschbarkeit technischer Produkte, die Unterstützung von Anforderungen und Funktionen durch die Produkte und die Stabilität von Produkten bewertet. Darüber hinaus wurde auch bewertet, ob Mitarbeiter Know-how bezüglich bestimmter Produkte haben bzw., welches Know-how aufgebaut werden müsste und mit welchem Aufwand dies verbunden ist.

Frage 1b: Welche Qualitätsattribute haben Sie betrachtet?

Die Qualitätsmodelle der Interviewpartner, die sich aus den Interviews ergaben, sind teilweise sehr unterschiedlich. Alle Qualitätsattribute der Interviewpartner sind in Tab. 6-1 aufgelistet. Ggf. werden Erläuterungen zu einzelnen Qualitätsattributen gegeben. In der Analyse der Qualitätsmodelle der Interviewpartner (Kapitel 6.2.1) wird dargestellt, welche Qualitätsattribute sie beinhalten.

Qualitätsattribut[1218]	Anzahl	Bemerkung
Allgemeingültigkeit	1	

[1217] Unter der Schönrechnungsgefahr sieht der Interviewpartner die Gefahr, dass viele Kriterien, die man in eine ROI-Bewertung hinzuzählen könnte, monetär nur schwer festlegbar sind, und eine Bewertung dann entsprechend der Ziele des Bewerters verfälscht wird.

[1218] Kursiv dargestellte Qualitätsattribute sind Qualitätsattribute des Modells dieser Arbeit.

Qualitätsattribut[1218]	Anzahl	Bemerkung
Änderbarkeit	4	
Antwortzeitverhalten	1	
Benutzbarkeit	2	Ein Interviewpartner nannte die Benutzbarkeit „Akzeptanz und Beherrschbarkeit durch Softwareentwickler“
Effizienz	3	Zwei Interviewpartner bewerteten die „Performanz“, die in dieser Arbeit dem Qualitätsattribut Effizienz zugeordnet wird.[1219]
Erweiterbarkeit	4	
Wandlungsfähigkeit	2	Der Wandlungsfähigkeit wurden in dieser Arbeit die Begriffe „Flexibilität“ und „Wartbarkeit“ zugerechnet.[1220]
Funktionserfüllung	1	
Geschäftsprozesssteuerung	2	Ein Interviewpartner stellte dar, dass die Geschäftsprozesssteuerung durch die „Zentralisierung der Business Logik“ möglich wird. Die Fähigkeit zur Geschäftsprozesssteuerung wird in dieser Arbeit als Bestandteil der Wandlungsfähigkeit angesehen.
Hard- & Softwareunabhängigkeit	1	
Infrastrukturfähigkeit	2	Die Infrastrukturfähigkeit einer Softwarearchitektur kann nur anhand spezifischer Situationen bewertet werden. Dies begründet sich darin,

[1219] Vgl. Kapitel 3.1.3.6.1.

[1220] Dies wurde vorgenommen, um die Interviewergebnisse mit dem Qualitätsmodell dieser Arbeit in Einklang zu bringen. Vgl. Kapitel 3.1.3.3.

Qualitätsattribut[1218]	Anzahl	Bemerkung
		weil die Infrastrutkturfähigkeit von den eingesetzten technischen Lösungen abhängt.[1221]
Installationsfähigkeit	1	Die Installationsfähigkeit wurde bezüglich einer spezifischen technischen Lösung (Linux) untersucht.
Integrationsgegenstand	2	
Integrationsreichweite	3	
Integrationsrichtung	1	
Integriertheit	2	
Interoperabilität	1	
Kommunikations-fähigkeit	1	
Kopplung	2	Der Grad der Kopplung ist eine Eigenschaft einer SOA.[1222] Die Ausprägungen von Eigenschaften einer Softwarearchitektur haben Einfluss auf Qualitätsattribute und dürfen deswegen nicht als Qualitätsattribut bewertet werden.
Modularität	1	
Nachhaltigkeit	2	
Prozessoptimierung	1	
Ressourcenausnutzung	1	
Skalierbarkeit	2	

[1221] Aus theoretischer Perspektive ist jede Softwarearchitektur geeignet eine Infrastruktur zu bilden, weil sie definiert, wie Softwarekomponenten gestaltet werden sollen und wie die Beziehungen dieser zueinander gestaltet sind (vgl. Kapitel 2.1). Spezielle technische Lösungen, um eine Softwarearchitektur für eine Unternehmens-IT einsetzen zu können, schränken die Infrastrukturfähigkeit jedoch u. U. ein.

[1222] Vgl. Kapitel 2.2.4.3.

Qualitätsattribut[1218]	Anzahl	Bemerkung
Stabilität	1	
Strategieunterstützung	1	Der Interviewpartner stellte dar, dass die Bewertung der Strategieunterstützung bisher noch nicht durchgeführt wurde, dies jedoch nachgeholt werden sollte.
Testbarkeit	3	Einer der Interviewpartner bewertete die Fehlerlokalisierung, die Teil der Testbarkeit ist.
Verfügbarkeit	2	
Verlässlichkeit	3	Von zwei Interviewpartnern wurde das Qualitätsattribut Zuverlässigkeit genannt.
Verständlichkeit	3	
Wiederverwendbarkeit	4	

Tab. 6-1: Qualitätsattribute der Interviewpartner

Frage 1c: Haben Sie technisch-fachliche und geschäftliche Qualitätsattribute unterschieden?

Die Antwort dieser Frage wurde meistens implizit während der Beantwortung der beiden vorhergehenden Fragen gegeben. Drei der Interviewpartner haben beide Kategorien von Qualitätsattributen beachtet. Einem weiteren fiel bei expliziter Fragestellung auf, dass auch die Strategieunterstützung beachtet werden sollte, dies jedoch von ihm nicht durchgeführt wurde.

Frage 1d: In welchen Bereichen haben Sie Kostenpotentiale identifiziert?

Kosten identifizierten alle Interviewpartner. Einer identifizierte Lizenzkosten und Kosten für Schnittstellen als Hauptkostenträger und stellte fest, dass ein Zuordnungsproblem von Kosten auf fachliche oder auf technische Kosten existiert. Ein anderer Interviewpartner identifizierte Kosten für Software und Lizenzen, Hardware, Personalaufwand für IT- und Fachabteilung sowie organisatorische Anpassungen. Diese Betrachtung wurde für einen Zeitraum von fünf Jahren vorgenommen. Ein dritter Inter-

viewpartner unterschied zwischen variablen Kosten und Fixkosten. Einer der Interviewpartner beschrieb den Vergleich mit Opportunitätskosten und Kosten auf Seiten des Fachbereichs. Der fünfte Interviewpartner identifizierte Personalkosten, Aufwand zur IT-Strategieänderung, Hardwarekosten, Lizenzkosten und Kosten zur Schnittstellenentwicklung. Dieser Interviewpartner betonte, dass Kosten der Fachseite, der Infrastruktur und der Qualifizierung nicht erhoben wurden.

Weil viele Daten bezüglich der identifizierten Kosten schon in Frage 1a beantwortet wurden, wird an dieser Stelle auf diese Daten verwiesen.

Frage 1e: Wie viel Aufwand haben Sie in die Bewertung investiert?

Die Angaben schwankten um 25 Personentage (PT). Ein Interviewpartner gab „ca. 25 PT“ an und ergänzte, dass die Angemessenheit des Aufwands stark von den Anforderungen des Managements abhängt: Fordert das Management detaillierte Bewertungen, kann der Aufwand schnell ansteigen. Ein anderer Interviewpartner investierte 21 PT – exklusive die Zeit von Auskunft gebenden Personen – in eine Bewertung. Ein Interviewpartner unterschied in Aufwand zur technischen Analyse, die mit 10-14 PT, und in Aufwand zur fachlichen Analyse, die mit ca. 5-20 PT, abhängig von der Größe des Projektes, zu Buche schlug. 60 PT investierte ein weiterer Interviewpartner. Der letzte Interviewpartner investierte ca. 20-30 PT, die Bewertung fand als begleitender Prozess statt, während schon mit dem Projekt begonnen wurde. In eine „Basisanalyse“ wurden hier 5 PT und in darauf folgende Tätigkeiten zur Organisation von Daten und deren Analyse weitere 25 PT investiert.

Frage 1f: Ist der betriebene Aufwand Ihrer Meinung nach angemessen?

Nach Auskunft von drei Interviewpartnern hängt der Aufwand von der Projektgröße bzw. dem geforderten Detaillierungsgrad ab und in den von diesen Interviewpartnern genannten Größenordnungen (gemittelt ungefähr 25 Tage) angemessen. Einer der Interviewpartner verwies darauf, dass die entstandenen Kosten der Bewertung gemessen an dem Gesamtaufwand einer Einführung einer SOA auf jeden Fall angemessen sind. (Dieser Interviewpartner hatte PT am oberen Ende der Angaben investiert.) Einer der Interviewpartner wollte darauf keine Antwort geben.

6.1.2.2 Fragenkomplex 2: Fragen zum Bewertungsergebnis

Die beiden Fragen des Fragenkomplex 2 sind: „Was war das Ergebnis der Bewertung?“ (2a) und „Konnte das Ergebnis verifiziert werden?“ (2b).

Einer der Interviewpartner, der verschiedene Business Cases durchgerechnet hat, kam zu dem Ergebnis, dass diese Bewertung teilweise positiv ist, jedoch nicht 100%ig abgesichert werden konnte, dass sie korrekt ist. Die Ergebnisse konnten im Nachhinein jedoch immer bestätigt werden.

Eine unterschiedliche Bewertung stellte ein anderer Interviewpartner fest. Der pauschalen Aussage, dass ein Architekturwechsel vorteilhaft sei, widersprach er. In Bewertungen, auf deren Grundlage eine Einführung beschlossen wurde, konnten diese verifiziert werden, wobei immer Abweichungen festgestellt wurden, die auf Änderungen bei Vorgehen gegenüber der bewerteten Situation zurückgeführt wurden.

Ein dritter Interviewpartner stellte fest, dass Voraussetzungen im betrachteten Projekt zur Einführung einer SOA nicht gegeben waren, weshalb eine Einführung nicht durchgeführt wurde. Ein abschließendes Ergebnis wäre vermutlich positiv gewesen, weil das Projekt jedoch gestoppt wurde, wurde die Bewertung abgebrochen und eine Verifizierung konnte nicht stattfinden.

Ein weiterer Interviewpartner stellte ebenso fest, dass Ergebnisse nicht pauschal ausgegeben werden können. Seine Bewertungen waren sowohl positiv als auch negativ. Eine Verifizierung konnte noch nicht vorgenommen werden.

Auch der fünfte Interviewpartner konnte Ergebnisse nicht verifizieren. Die Ergebnisse seiner Untersuchungen waren auch unterschiedlich, in den meisten Fällen jedoch derart ausgeprägt, dass bestehende Probleme eines Unternehmens durch eine Einführung einer SOA behoben werden konnten.

6.1.2.3 Fragenkomplex 3: Fragen zum Interviewpartner

Der Fragenkomplex drei dient dazu abzusichern, dass die Interviewpartner qualifizierte Aussagen machen können und die erhobenen Daten geeignet sind, die aktuelle Praxis der Architekturbewertung und insbesondere die SOA-Bewertung abzubilden. Die gestellten Fragen sind: „Wie viele SOA-Projekte haben Sie bewertet?“ (3a) (Hat der Interviewpartner kein SOA-Projekt bewertet, wurde nach Softwarearchitekturen gefragt.)

„Welchen Umfang (in Personentagen) hatten diese Projekte?“ (3b) und „Wann fanden diese Bewertungen statt?“ (3c).

Von den fünf Interviewpartnern haben drei insgesamt sechs SOA-Bewertungen vorgenommen. Die anderen beiden Interviewpartner haben sieben Softwarearchitektur-Bewertungen durchgeführt. Ein Interviewpartner ergänzte noch Erfahrungen im Bereich der Bewertung „kleinerer Integrationsprojekte“, die in der Menge im zweistelligen Bereich liegen.

Bezüglich des Umfangs der bewerteten Projekte wollten drei der fünf Befragten keine konkrete Aussage abgeben. Zwei Interviewpartner haben grobe Größen angegeben: ein Projekt war im großen zweistelligen Millionenbereich (Euro) angesiedelt, zwei weitere Projekte im einstelligen Millionenbereich und ein weiteres umfasst ca. 30.000 Personentage.

Die Bewertungen der SOA-Projekte fanden im Zeitraum von 2002 bis 2005 statt, mit einem Schwerpunkt von drei Untersuchungen in 2003. Die jüngsten Bewertungen waren zwei aus dem Jahr 2005 (Frühjahr bzw. Sommer). Die älteste begann schon im Jahr 2002 mit einer Vorstudie, die Hauptuntersuchung jedoch fand ab 2003 statt.

6.2 Analyse und Diskussion

Die Analyse und Diskussion der erhobenen Daten wird anhand der Kriterien des Qualitätsmodells (Kapitel 6.2.1), einer Kostenanalyse (Kapitel 6.2.2) und einer Wirtschaftlichkeitsanalyse (Kapitel 6.2.3) geführt. Anschließend wird die Eignung der Interviewpartner untersucht (Kapitel 6.2.4).

6.2.1 Qualitätsmodell

Ein Qualitätsmodell zeichnet sich durch die verwendeten Qualitätsattribute aus. Bezüglich der Qualitätsattribute zeigt sich, dass

1. das Bewertungsmodell immer nur auszugsweise verwendet wurde,
2. zur Architekturbewertung nicht relevante Qualitätsattribute bewertet wurden,
3. über das Bewertungsmodell dieser Arbeit hinausgehend keine Qualitätsattribute identifiziert wurden und

4. sowohl technisch-fachliche als auch geschäftliche Qualitätsattribute bewertet wurden.

Auszugsweise Nutzung des Bewertungsmodells

Die fünf Interviewpartner wenden verschiedene Qualitätsmodelle an, um eine SOA zu bewerten. Es zeigt sich, dass alle Interviewpartner nur Teile des in dieser Arbeit aufgestellten Qualitätsmodells verwenden.

Der erste[1223] Interviewpartner bewertete anhand von Qualitätsattributen, die zwei Qualitäts- und fünf Unterqualitätsattribute in dieser Arbeit darstellen. Die verwendeten Qualitätsattribute sind Wiederverwendbarkeit und Verlässlichkeit; die verwendeten Unterqualitätsattribute sind Allgemeingültigkeit, Änderbarkeit, Erweiterbarkeit, Testbarkeit und Verständlichkeit. Mit den Unterqualitätsattributen Änderbarkeit, Erweiterbarkeit und Testbarkeit deckt der Interviewpartner einen Teil des Qualitätsattributes Wandlungsfähigkeit ab.[1224] Verständlichkeit ist ein Teil der Benutzbarkeit. Die Allgemeingültigkeit ist ein Teil der Wiederverwendbarkeit, die als separates Qualitätsattribut genannt wurde. Über diese Qualitätsattribute hinaus wurden noch zwei Qualitätsattribute genannt, die für eine von technischen Lösungen abstrahierende Bewertung irrelevant sind: Software und Hardware-Unabhängigkeit.

Der zweite Interviewpartner bewertete anhand von Qualitätsattributen, die drei Qualitäts- und vier Unterqualitätsattribute in dieser Arbeit darstellen. Die verwendeten Qualitätsattribute sind Funktionserfüllung, Effizienz und Strategieunterstützung; die verwendeten Unterqualitätsattribute sind Änderbarkeit, Erweiterbarkeit, Integrationsgegenstand und Integrationsreichweite. Die beiden Unterqualitätsattribute Änderbarkeit und Erweiterbarkeit sind ein Teil der Unterqualitätsattribute der Wandlungsfähigkeit. Integrationsgegenstand und Integrationsreichweite sind ein Teil der Unterqualitätsattribute der Integriertheit. Die Strategieunterstützung wurde von dem Interviewpartner nicht bewertet, er stellte jedoch heraus, dass eine solche Bewertung stattfinden müsste. Weil er sich dessen bewusst ist, wird das Qualitätsattribut als ein Bestandteil seines Qualitätsmodells aufgenommen.

[1223] Um die Anonymitätsanforderungen der Interviewpartner zu erfüllen, ist die Reihenfolge hier zufällig festgelegt.

[1224] Vgl. Kapitel 3.1.3.3.2.

Der dritte Interviewpartner bewertete anhand von Qualitätsattributen, die drei Qualitäts- und sieben Unterqualitätsattribute in dieser Arbeit darstellen. Die verwendeten Qualitätsattribute sind Verlässlichkeit, Wiederverwendbarkeit und Nachhaltigkeit; die verwendeten Unterqualitätsattribute sind Änderbarkeit, Erweiterbarkeit, Verständlichkeit, Kommunikationsfähigkeit, Verfügbarkeit, Modularität und Integrationsreichweite. Die beiden Unterqualitätsattribute Änderbarkeit und Erweiterbarkeit sind ein Teil der Unterqualitätsattribute der Wandlungsfähigkeit. Verständlichkeit und Kommunikationsfähigkeit sind ein Teil der Unterqualitätsattribute der Benutzbarkeit. Die Integrationsreichweite ist ein Teil der Unterqualitätsattribute der Integriertheit. Das Unterqualitätsattribut Verfügbarkeit ist in dieser Arbeit ein Teil der Unterqualitätsattribute der Verlässlichkeit, und das Unterqualitätsattribut Modularität ist in dieser Arbeit ein Teil der Unterqualitätsattribute der Wiederverwendbarkeit. Beide Qualitätsattribute wurden von dem Interviewpartner auch separat bewertet.

Der vierte Interviewpartner bewertete anhand von Qualitätsattributen, die vier Qualitäts- und drei Unterqualitätsattribute in dieser Arbeit darstellen. Die verwendeten Qualitätsattribute sind Wandlungsfähigkeit (Wartbarkeit[1225]), Effizienz, Wiederverwendbarkeit und Integriertheit (Integrationsform[1226]); die verwendeten Unterqualitätsattribute sind Testbarkeit, Verständlichkeit (Durchgängigkeit[1227]) und Ressourcenausnutzung. Das Unterqualitätsattribut Testbarkeit ist in dieser Arbeit ein Teil der Unterqualitätsattribute der Wandlungsfähigkeit, die von dem Interviewpartner auch separat bewertet wurde. Der Grad der Ressourcenausnutzung als Unterqualitätsattribut ist ein Bestandteil der Effizienz – auch dies wurde beides separat bewertet.

Der fünfte Interviewpartner bewertete anhand von Qualitätsattributen, die sechs Qualitäts- und elf Unterqualitätsattribute in dieser Arbeit darstellen. Die verwendeten Qualitätsattribute sind Funktionserfüllung, Benutzbarkeit, Verlässlichkeit, Effizienz, Wiederverwendbarkeit, Nachhaltigkeit und Integriertheit; die verwendeten Unterqualitätsattri-

---

[1225] Der Interviewpartner nannte Wartbarkeit. Dieser Begriff wird in dieser Arbeit als ‚Wandlungsfähigkeit' bezeichnet.

[1226] Die Integrationsform definiert für den Interviewpartner die Art, wie integriert wird und was dadurch erreicht werden kann. Durch diese allgemeine Darstellung entspricht dieses Qualitätsattribut allen Unterqualitätsattributen der Integriertheit: dem Intergrationsgegenstand, der -reichweite und der -richtung.

[1227] Der Interviewpartner nannte Durchgängigkeit als Qualitätsattribut. Nachfragen führten zu dem Ergebnis, dass der Interviewpartner damit die ‚Verständlichkeit' im Sinne dieser Arbeit auffasst und z. B. nicht ‚Nachhaltigkeit'.

bute sind Änderbarkeit, Testbarkeit, Erweiterbarkeit, Stabilität, Verfügbarkeit, Antwortzeitverhalten, Installationsfähigkeit, Integrationsgegenstand, Integrationsreichweite und Integrationsrichtung. Die Unterqualitätsattribute Änderbarkeit, Testbarkeit, Erweiterbarkeit und Stabilität sind alle Unterqualitätsattribute der (durch den Interviewpartner nicht explizit bewerteten) Wandlungsfähigkeit dieser Arbeit. Auch können alle Unterqualitätsattribute der Integriertheit im Qualitätsmodell gefunden werden. Verfügbarkeit ist ein Teil der Verlässlichkeit, die vom Interviewpartner auch separat bewertet wurde. Das Antwortzeitverhalten ist als Unterqualitätsattribut ein Bestandteil der Effizienz – beides wurde ebenso separat bewertet. Die Installationsfähigkeit wurde durch den Interviewpartner für die spezielle technische Lösung ‚Linux' untersucht und bewertet. Dies wird für diese Analyse als Bestandteil des Qualitätsmodells gewertet.

Tab. 6-2 stellt zusammenfassend dar, anhand wie vieler (Unter-)Qualitätsattribute durch die Interviewpartner bewertet wurde. Dargestellt wird auch eine Annahme für den bestmöglichen Fall (im Folgenden Best-Case-Annahme genannt). Diese beschreibt, wie viele Qualitätsattribute durch den Interviewpartner untersucht wurden, wenn durch die Bewertung der (im jeweiligen Qualitätsmodell vorhandenen) Unterqualitätsattribute eine ausreichende Bewertung der entsprechenden Qualitätsattribute erreicht wird. Im Qualitätsmodell eines Interviewpartners sind z. B. die Unterqualitätsattribute Änderbarkeit und Erweiterbarkeit vorhanden. Die Best-Case-Annahme unterstellt nun, dass die Wandlungsfähigkeit anhand dieser Qualitätsattribute ein qualitativ vergleichbares Ergebnis erreicht, wie im Vergleich dazu, dass zusätzlich auch die Unterqualitätsattribute Testbarkeit und Stabilität (des Modells dieser Arbeit) bewertet wurden.

Anhand Tab. 6-2 ist ersichtlich, dass die Interviewpartner keineswegs das gesamte Modell zur Bewertung der Wirtschaftlichkeit einer Softwarearchitektur in jeweiligen Bewertungen herangezogen haben. Dadurch können ausschlaggebende Nutzenpotentiale u. U. unbeachtet bleiben und eine verzerrte Bewertung des Nutzenpotentials kann angenommen werden. Verstärkt wird dies noch durch die fehlende Unterscheidung in Qualitätsattribute und Unterqualitätsattribute. Dies zeigt, dass eine Veränderung des Gewichts eines Qualitätsattributes unbeabsichtigt vorgenommen wurde, weil Unterqualitätsattribute zusätzlich bewertet wurden. Es weist darauf hin, dass praktischen Anwendern oftmals unklar ist, dass eine detaillierte Aufteilung und Bewertung stattfinden kann.

Interviewpartner	Qualitäts-attribute	Unterqualitäts-attribute	Best-Case-Annahme
Eins	2	4	4
Zwei	3	4	5
Drei	3	7	6
Vier	4	3	5
Fünf	6	11	8
(Diese Arbeit)	10	32	10

Tab. 6-2: Auswertung der verwendeten Qualitätsattribute

Eine Auswertung aller Qualitätsattribute zeigt, dass ein Qualitätsattribut von keinem Interviewpartner bewertet wurde: die Portabilität. Einzig ein Interviewpartner bewertete ein Unterqualitätsattribut der Portabilität: die Installationsfähigkeit.[1228] Klammert man die Bemerkung des zweiten Interviewpartners bezüglich der Strategieunterstützung aus, wurde diese ebenfalls von keinem Interviewpartner bewertet. Bezüglich der Unterqualitätsattribute kann festgestellt werden, dass 13 von 32 Unterqualitätsattributen von keinem und weitere zwölf von nur einem Interviewpartner bewertet wurden.

Würden alle Qualitätsmodelle der Interviewpartner zu einem aggregiert werden, würde dieses Qualitätsmodell immerhin 90 % der Qualitätsattribute und ca. 60 % der Unterqualitätsattribute des Qualitätsmodells dieser Arbeit umfassen. Eine vollständige Abdeckung, also eine Beachtung aller Unterqualitätsattribute dieser Arbeit, würde in diesem Fall immerhin bezüglich der Wandlungsfähigkeit, der Strategieunterstützung und der Integriertheit erreicht werden. Nur im Fall der Effizienz und der Wiederverwendbarkeit würde ein Unterqualitätsattribut unbeachtet bleiben, bei den anderen unvollständig beachteten Qualitätsattributen fehlen zwei bzw. drei Unterqualitätsattribute. Die durchschnittliche Abdeckung der Unterqualitätsattribute in einem aggregierten Qualitätsmodell würde ca. 57 % gegenüber dem Modell dieser Arbeit betragen.

[1228] Dies jedoch nur in Bezug auf eine spezifische technische Lösung – das Betriebssystem Linux.

Verwendung irrelevanter Qualitätsattribute

Auch irrelevante Qualitätsattribute im Sinne dieser Arbeit wurden in den Qualitätsmodellen der Interviewpartner verwendet. Diese sind Qualitätsattribute, die einen reinen technischen Bezug aufweisen, z. B. gehört hierzu die ‚Hardwareunabhängigkeit'.

Eine Erklärung für die Verwendung von Qualitätsattributen, die in dieser Arbeit als nicht relevant erachtet wurden, sind die Rahmenbedingungen, denen diese Arbeit unterliegt. In der Praxis können Situationen und Anforderungen vorliegen, sodass eine Betrachtung irrelevanter Qualitätsattribute im Sinne dieser Arbeit sinnvoll oder sogar notwendig ist. Beispielsweise kann die Softwareunabhängigkeit für ein Unternehmen extrem wichtig werden, wenn die Umsetzungsfähigkeit einer SOA aufgrund einer inhomogenen Softwareumgebung im Unternehmen bei gleichzeitiger Softwareabhängigkeit gewählter technischer Lösungen gefährdet ist.

Keine Verwendung zusätzlicher relevanter Qualitätsattribute

Die Interviewpartner verwendeten in ihren Qualitätsmodellen keine Qualitätsattribute, die zur Softwarearchitekturbewertung relevant und nicht im Qualitätsmodell dieser Arbeit vorhanden sind.

Verwendung technisch-fachlicher als auch geschäftlicher Qualitätsattribute

Vier der fünf Interviewpartner verwenden in ihren Qualitätsmodellen mindestens ein Qualitätsattribut der geschäftlichen Qualitätsattribute dieser Arbeit. Dies bestätigt, dass die Bewertung einer Softwarearchitektur um die Bewertung geschäftlicher Kriterien erweitert werden muss.

6.2.2 Analyse spezifischer SOA-Kosten

Spezifische Kostenpotentiale für die Verwendung einer SOA haben vier Interviewpartner untersucht.[1229] Spezifische Kosten einer SOA, die durch diese vier Interviewpartner identifiziert wurden, sind:

- Anschaffungs- und Lizenzkosten für Software und Hardware (durch drei Interviewpartner identifiziert und bewertet)

[1229] Vgl. Kapitel 6.1.2.1.

- Aufwand für Entwicklung und Wartung (z. B. Betriebskosten, Schnittstellen einer SOA[1230] und Personalaufwand) (drei Interviewpartner)
- Kosten durch organisatorische Anpassungen (ein Interviewpartner)
- Kosten der Systemablösung (ein Interviewpartner)
- Kosten des Managements einer SOA (ein Interviewpartner)

Im zusammengefassten Kostenmodell der Interviewpartner wurden die gleichen Kostenpotentiale herangezogen, die in dieser Arbeit Verwendung finden. Jedoch ergeben sich signifikante Unterschiede bezüglich der Ausarbeitungen: Die Interviewpartner untersuchten immer spezifische quantitative Ausprägungen, beispielsweise die anfallenden Betriebskosten und die Kosten zur Entwicklung von Schnittstellen. In dieser Arbeit dagegen kann aufgrund der gezogenen Rahmenbedingungen[1231] nur eine qualitative Untersuchung durchgeführt werden. Die Interviewpartner haben demnach jeweils einzelne Kostenpotentiale des Kostenmodells dieser Arbeit zur Bewertung herangezogen und für den konkreten Fall bewertet.

Einer der Interviewpartner identifizierte Kosten anhand von „Business Case“-Studien für spezifische Fragestellungen. In einem „Business Case“ wurden „Opportunitätskosten“ für den Einsatz in einer spezifischen Fachabteilung berechnet. Diese Berechnung jedoch betrachtete die potentiell neu zu erstellende Anwendung auf Basis einer SOA gegenüber der bestehenden Anwendung. Sie bewertete keine spezifischen Kosten der Softwarearchitektur, sondern die Kostenauswirkungen der neu zu erstellenden Anwendung.[1232] Deswegen ist dieses Kostenkriterium für das Kostenmodell dieser Arbeit nicht relevant.

Ein Interviewpartner untersuchte die Kosten für alternative Granularitätsstufen der zu verwendenden Schnittstellen (Kostenpotential ‚Entwicklung von Schnittstellen’). Dabei wurden grob- und feingranulare Schnittstellen unterschieden. Die Bewertung kam zu dem Ergebnis, dass feingranulare Schnittstellen die 3,5fachen Kosten verursachen gegenüber den Grobgranularen.

[1230] Einer der Interviewpartner kam hier zu dem Schluss, dass Schnittstellen erst dann mittels einer SOA günstiger werden, sobald sehr viele in einer Unternehmens-IT eingesetzt werden müssen.

[1231] Vgl. Kapitel 1.3.3.

[1232] In diesem Business-Case wurde der Zeitbedarf in der Anwendung der bestehenden gegenüber der neu zu erstellenden Anwendung monetär bewertet.

Zusammengenommen weisen die Interviewpartner ein gutes Verständnis für spezifische Kostenpotentiale einer SOA auf. Jedoch hat kein Interviewpartner mehr als drei Kostenpotentiale erkannt. Auffällig ist die Fokussierung auf Kosten der Infrastruktur (z. B. Soft- und Hardware) und auf Kosten der Entwicklung und des Betriebs einer SOA. Diese Kostenpotentiale fallen bei dem Einsatz einer Unternehmens-IT immer an. Dass jeweils zwei Interviewpartner – es handelt sich hierbei nicht um die gleichen – diese Kostenpotentiale nicht untersuchen erscheint verwunderlich. Ein Interviewpartner gab eine Erklärung hierfür ab: Er hat eine „Architekturbewertung“ einer SOA vorgenommen, indem er Nutzenpotentiale bewertete. Die Wirtschaftlichkeit habe er „fast nicht“ untersucht. Dass zwei andere Interviewpartner jedoch jeweils einen dieser beiden Kostenpotentiale nicht bewertet haben bleibt unerklärt. Dies deutet darauf hin, dass bei der Identifizierung relevanter Kostenpotentiale in der Praxis (1) entweder unklar ist, was bewertet werden soll, oder (2) bestimmte Bewertungen nicht durchgeführt werden können. Der zweite Fall kann jedoch ausgeschlossen werden, weil die Interviewpartner nach allen Kostenpotentialen gefragt wurden, auch nach denen, die nicht bewertet werden konnten. Aus diesem Grund muss angenommen werden, dass den praktischen Anwendern unklar ist, welche Kostenpotentiale zur SOA-Bewertung verwendet werden sollten.

6.2.3 Wirtschaftlichkeitsanalyse

Bezüglich der Wirtschaftlichkeitsanalyse zeigt sich, dass die Wirtschaftlichkeit gemäß dem Modell dieser Arbeit niemals vollständig bewertet wurde.

Die Wirtschaftlichkeit wurde anhand von ROI- und Kosten-Nutzen-Bewertungen vorgenommen. Für eine SOA-Bewertung verwendete ein Interviewpartner ein „erweitertes Kostenmodell“ und stellte darüber eine „Zusatzkostenbewertung“ auf. Ein anderer Interviewpartner orientierte sich zur ROI-Bewertung an „Business Cases“ und betrachtete das SOA-Projekt wie „jedes andere Projekt.“ Der Interviewpartner bewertete in diesem Fall den Umsatz (z. B. mit Drittparteien wie Kunden und Lieferanten) und die Gewinnentwicklung auf einer festgelegten Zeitschiene („Net Revenues“, herangezogen wurden Kennzahlen wie variable Kosten und Fixkosten).

Stellt man die vorgenommenen Bewertungen der Interviewpartner[1233] dem Wirtschaftlichkeitsmodell dieser Arbeit gegenüber, entsprechen die Bewertungen den Wirtschaft-

lichkeitspotentialen ‚Kundenzufriedenheit', ‚Beziehungen zu Geschäftspartnern' und ‚Interne Wirkungen'. Wirtschaftlichkeitspotentiale bezüglich der Mitarbeiterzufriedenheit und bezüglich des Wettbewerbs wurden von keinem Interviewpartner identifiziert. Folglich stützt sich die Bewertung der Wirtschaftlichkeit durch die Interviewpartner *nicht* auf alle relevanten Wirtschaftlichkeitspotentiale.

6.2.4 Eignung der Interviewpartner

Drei der fünf Interviewpartner haben jeweils mindestens eine Wirtschaftlichkeitsbewertung einer SOA vorgenommen. Die anderen Interviewpartner haben Bewertungen von Softwarearchitekturen durchgeführt. Die Erfahrungen und das erhobene Datenmaterial stammen aus den Jahren 2001-2005. Der Großteil der erhobenen Daten bezieht sich auf das Jahr 2003. Das Datenmaterial und die Erfahrungen der Interviewpartner sind deswegen aktuell und adäquat.

6.3 Zusammenfassung

Horváth beschrieb schon 1988, dass die Bewertung der Wirtschaftlichkeit anhand weniger Kriterien durchgeführt und dadurch eingeengt bzw. vergröbert wird.[1234] An diesem Zustand scheint sich in der Praxis nicht viel geändert zu haben. Diese Arbeit liefert einen signifikanten Beitrag zum Erkenntnisstand – auch für die Praxis. Substantielle Erkenntnisse insbesondere in der Bewertung des Nutzenpotentials und der Wirtschaftlichkeitsanalyse werden durch diese Arbeit übermittelt. Mit deren Hilfe kann die Praxis eine durchgängige Bewertung der Wirtschaftlichkeit einer SOA erstellen. Fasst man alle Kostenpotentiale der Interviewpartner zusammen haben die Interviewpartner alle relevanten Kostenpotentiale identifiziert. Einzeln betrachtet ergeben sich hier jedoch ebenfalls große Lücken, die mithilfe dieser Arbeit geschlossen werden können.

[1233] Vgl. Kapitel 6.1.2.1.

[1234] Vgl. Horváth /Grundprobleme/ S. 2-3.

7. Resümee und Ausblick

Unternehmen müssen heute große Anstrengungen bewerkstelligen, um eine Softwarearchitektur zu finden, die alle gestellten Herausforderungen bewältigt.[1235] Erstens, weil viele Softwarearchitekturen existieren – insbesondere, wenn man die technischen Unterschiede in die Auswahlentscheidung einbezieht; zweitens, weil es unklar ist, welche Auswirkungen der Einsatz verschiedener Softwarearchitekturen hat. Gerade hier wird in der Praxis selten erkannt, welche weit reichenden Auswirkungen Softwarearchitekturen auf die Wirtschaftlichkeit des gesamten Unternehmens haben.

Die klassische Investitionsrechnung muss nach Horváth zur Wirtschaftlichkeitsanalyse neuer Produktions- und Informationstechnologien erweitert werden, damit alle relevanten Kosten und Nutzenaspekte berücksichtigt werden.[1236] Schlechte IT-Investitionsentscheidungen, vor allem solche, die indirekte Kosten nicht beachten, können dazu führen, dass die Wettbewerbsfähigkeit abnimmt, finanzielle Rückflüsse zurückgehen oder Arbeitsplätze abgebaut werden müssen.[1237] Irani und Love haben festgestellt, dass Manager dazu neigen kurzsichtig zu entscheiden, wenn es um Investitionen in IT geht – primär deswegen, weil keine geeigneten Methoden zur Wirtschaftlichkeitsbewertung zur Verfügung stehen![1238] Die empirische Untersuchung hat gezeigt, dass der Praxis entsprechende Methoden und Modelle im notwendig gewordenen Umfang fehlen. Ebenso ist in der Theorie Wissen um Nutzen und Kosten verschiedener Softwarearchitekturen sowie das Wissen, anhand welcher Qualitätsattribute und Kriterien diese gemessen werden sollten, noch nicht ausreichend aufgebaut.[1239] Der bisherige Erkenntnisstand zur Bewertung von Softwarearchitekturen kann zweifelsfrei und trotz hoher Relevanz als gering angesehen werden.

In dieser Arbeit wurde (1) ein begründetes Qualitätsmodell zur Bewertung von Softwarearchitekturen und (2) ein Modell zur Wirtschaftlichkeitsanalyse für Softwarearchitekturen aufgestellt. Das Modell berücksichtigt, dass Softwarearchitekturen extrem langlebig sind und eine gesamte Unternehmens-IT betreffen. Deswegen müssen diese

---

[1235] Vgl. zu diesem Absatz MacCormack, Verganti, Iansiti /Products/ S. 136.

[1236] Vgl. Horváth /Grundprobleme/ S. 13.

[1237] Vgl. Alshawi, Irani, Baldwin /Benchmarking/ S. 415.

[1238] Vgl. Irani, Love /Propagation/ S. 163; Für weitere Argumentationen vgl. Alshawi, Irani, Baldwin /Benchmarking/ S. 416-417.

[1239] Vgl. Svahnberg, Wohlin /Quality Attributes/ S. 149.

anhand von Qualitätsattributen bewertet werden, die in dem hier vorgestellten Qualitätsmodell dargestellt sind. In dieser integrierten Zusammenstellung – unter der Beachtung von *technisch-fachlichen und geschäftlichen Qualitätsattributen* – sind Softwarearchitekturen noch nicht bewertet worden.

Das erstellte Modell wurde auf eine SOA angewendet. Für den speziellen Fall der Bewertung einer Investition in eine Softwarearchitektur, die nach Prinzipien einer SOA aufgebaut ist, stellt die vorliegende Arbeit einen sorgfältig ausgearbeiteten Rahmen zur Verfügung, anhand dessen eine Investitionsentscheidung auf eine fundierte Basis gestellt werden kann. Damit liefert sie eine Unterstützung zur Entscheidung über die Investition in eine SOA. Das Wirtschaftlichkeitsmodell und sogar das Qualitätsmodell selbst können von Praktikern zur Entscheidungsunterstützung und von Theoretikern zur weiteren Forschung verwendet werden.

7.1 Kritische Würdigung

Die Ziele der Arbeit konnten zwar erreicht werden, die erzielten Ergebnisse sind aber mit verschiedenen Einschränkungen und offen gebliebenen Fragen verbunden. Diese betreffen Rahmenbedingungen, das aufgestellte Qualitäts- und Wirtschaftlichkeitsmodell, die Argumentation und die Bewertung.

Rahmenbedingungen

Es wurde festgestellt, dass es schwierig ist, Investitionsentscheidungen in IT methodisch auf eine gefestigte Basis zu stellen.[1240] Diese Arbeit versucht dennoch dies zu erreichen. Damit ein Bewertungsrahmen erstellt werden konnte, welcher angemessen ist, mussten spezielle Rahmenbedingungen herangezogen werden. Für diese Arbeit sind die Rahmenbedingungen die Fokussierung auf die Bewertung von (1) einer idealtypischen SOA für (2) die Entscheidungssituation von Unternehmen. Es muss angenommen werden, dass einzelne Argumentationen oder Nutzenketten nicht für alle Unternehmen bzw. Situationen gelten. So könnten Unternehmen einer bestimmten Nische, welche besonderen Anforderungen ausgesetzt sind, einzelnen Argumenten widersprechen.[1241] Bei-

[1240] Vgl. Kapitel 1.2.

[1241] Alshawi, Irani und Baldwin z. B. stellen fest, dass Unternehmen mit gleichen IT-Systemen unterschiedlichen Nutzen aus ihrer IT haben, vgl. Alshawi, Irani, Baldwin /Benchmarking/ S. 419.

spielsweise könnten karitative Unternehmen oder militärische Einrichtungen andere grundlegende Ziele verfolgen als marktwirtschaftlich handelnde Unternehmen.

Eine weitere Rahmenbedingung dieser Arbeit stellt die Abstrahierung von technischen Lösungen und Produkten dar. Weil in der Praxis auf eine technische Lösung und auch auf den Einsatz von Produkten nicht verzichtet werden kann, könnten zur Bewertung einer konkreten Situation weitere, technik-affine Qualitätsattribute notwendig werden. Dadurch können weitere Kostenpotentiale entstehen. Außerdem ergeben sich Auswirkungen auf die Wirtschaftlichkeit einer SOA für den konkreten Fall und unter Beachtung von technischen Lösungen und Produkten. Eine Untersuchung, (1) welche technischen Lösungen und Produkte existieren und wie diese kombiniert werden können, (2) welche Erweiterungen der Bewertung (z. B. in Form von Qualitätsattributen) notwendig werden und (3) welche Auswirkungen diese auf die Wirtschaftlichkeitsrechung haben, wäre für diese Arbeit zu umfangreich geworden. Weitere Forschung bezüglich oben aufgezeigter technik-affiner Bewertung muss noch erfolgen.

Qualitäts- und Wirtschaftlichkeitsmodell

Frese kommt zu dem Schluss, dass Informationstechnologie und Unternehmensorganisation in keinem deterministischen Ursache-Wirkungs-Zusammenhang stehen.[1242] Daraus könnte die Anforderung abgeleitet werden, dass Aspekte der Unternehmensorganisation für die Bewertung einer Softwarearchitektur nicht relevant sind. Dem muss jedoch widersprochen werden und es konnte mehrfach gezeigt werden, dass die Art der Organisationsgestaltung durchaus Einfluss auf eine Unternehmens-IT haben kann – zumindest wird die entsprechende Wirtschaftlichkeit beeinflusst. Frese liegt dennoch richtig: Er betrachtet Informationstechnologie aus der rein technischen Perspektive. Ein deterministischer Ursache-Wirkungs-Zusammenhang wird sich diesem Verständnis nach nicht nachweisen lassen.

Technisch-fachliche Qualitätsattribute können über Standards und Standardarbeiten zu diesem Thema erhoben werden. Geschäftliche Qualitätsattribute wurden mittels eines fokussierten Literaturstudiums gesucht. Durch diese Vorgehensweise kann nicht garantiert werden, dass alle zur Architekturbewertung relevanten geschäftlichen Qualitätsattribute in dieser Arbeit untersucht und in das Qualitätsmodell aufgenommen wurden. Es

kann sich in der Zukunft durchaus zeigen, dass ein weiteres geschäftliches Qualitätsattribut zur Bewertung herangezogen werden muss.[1243]

Das entstandene Qualitätsmodell mag umfangreich erscheinen. Die Wirtschaftlichkeit kann jedoch nur dann bewertet werden, wenn alle relevanten Aspekte bewertet werden.[1244] Beispielsweise müssen konkurrierende Sachverhalte berücksichtigt werden, oder es darf keine Einschränkung auf nur eine Kostenart vorliegen. Kriterien zur Bewertung sind zielbezogen und problemindividuell festzulegen. Beides wurde in dieser Arbeit erreicht. Ein zielbezogenes und problemindividuelles Qualitätsmodell wurde erstellt: Es bezieht sich auf Softwarearchitekturen. Die Bewertung der Wirtschaftlichkeit wurde ebenso zielbezogen und problemindividuell vorgenommen: (1) Unterschiedliche Nutzergruppen wurden identifiziert; (2) Faktoren einer Softwarearchitektur, die auf diese Nutzergruppen Einfluss haben, wurden identifiziert und (3) eine Bewertung dieser Zusammenhänge für eine spezifische Ausprägung einer Softwarearchitektur, eine SOA, wurde vorgenommen.

Argumentation und Bewertung

Die in dieser Arbeit getätigten Argumentationen beruhen auf der Literatur, Erkenntnissen aus verschiedenen Diplomarbeiten und Diskussionen im Arbeitskreis SOA der Gesellschaft für Informatik e. V. zu dem Thema SOA. Die Aussagen mussten allgemein gehalten werden und können für spezifische Praxisfälle zu dieser Arbeit verschieden ausgeprägt sein. Für solche Praxisfälle, z. B. die Bewertung der Einführung einer SOA in einem bestimmten Unternehmen zur Ablösung festgelegter IT-Systeme, können die entsprechenden Bewertungen sogar in Kardinalskalen angegeben werden.

Eine tautologische Argumentation liegt in dieser Arbeit nicht vor. Das Risiko einer tautologischen Argumentation ergibt sich, wenn eine Annahme aufgestellt wird, darauf aufbauend argumentiert wird, dass eine bestimmte Nutzeffektkette vorliegt und schließlich zu dem Ergebnis kommt, dass die Annahme erfüllt ist. Bei den folgenden Argumentationen wird häufig darauf hingewiesen, dass bestimmte Gestaltungsprinzipien

---

[1242] Vgl. Frese /Organisation/ S. 142.

[1243] Kieser und Walgenbach diskutieren die Notwendigkeit der Erweiterung von Bewertungsmodellen um weitere Bewertungsdimensionen beispielhaft für ein Bewertungsmodell von Organisationsstrukturen. Vgl. Kieser, Walgenbach /Organisation/ S. 71-76.

[1244] Vgl. zum folgenden Absatz Horváth /Grundprobleme/ S. 2-3.

eingehalten und bestimmte Gestaltungsziele verfolgt werden sollten, damit eine SOA vorliegt und darauf aufbauend werden Nutzenketten und Argumentenbilanzen aufgestellt. Jedoch wird dadurch nicht belegt, dass die aufgestellten Gestaltungsprinzipien erfüllt werden, sondern vielmehr, dass die Verfolgung dieser – also die Verwendung einer SOA – Nutzen bringt. Softwarearchitekturen beruhen darauf, dass solche Gestaltungsprinzipien und –ziele verfolgt werden, weil sich Softwarearchitekturen durch entsprechende Gestaltungsprinzipien manifestieren.[1245]

7.2 Verwendungsmöglichkeiten

Die Bewertung von Softwarearchitekturen erscheint nach dem Studium dieser Arbeit aufwendig. Posch, Birken und Gerdom argumentieren jedoch, dass sich eine Architekturbewertung durchaus wirtschaftlich lohnt.[1246] Das aufgestellte Wirtschaftlichkeitsmodell eignet sich zur Anwendung in der Praxis. Es liefert Ansatzpunkte, anhand deren die Wirtschaftlichkeit von Investitionen in IT bewertet werden können. Das Qualitätsmodell alleine stehend kann z. B. auch zur Auswahl einer Technologie für Infrastrukturplattformen verwendet werden.

Auch für die Theorie ist die Arbeit wertvoll. Aufbauend auf der Arbeit können Folgearbeiten aufgesetzt werden, die das Wirkungsmodell, z. B. durch Fallstudien, validieren und seine praktische Anwendbarkeit demonstrieren. Möglich wäre der Vergleich von Bewertungsergebnissen mit abgeschlossenen bewertbaren Projekten. Dadurch könnte eine Validierung des Modells erreicht werden. Momentan erscheint es schwer möglich, weil keine geeigneten und bewerteten Untersuchungsgegenstände – in diesem Fall die Implementation einer SOA im Unternehmen – verfügbar sind. Auch müssten Untersuchungen vorgenommen werden, die ausschließen, dass Mängel an abgeschlossenen Projekten andere Ursachen haben, z. B. mangelnder Wissensstand, um eine SOA zu implementieren.

[1245] Z. B. kann auch mit komponentenorientierter Technologie wie CORBA eine Großrechnerarchitektur aufgebaut werden.

[1246] Vgl. Posch, Birken, Gerdom /Basiswissen/ S. 195.

Carr hat provokant aufgezeigt, dass IT heute kein Unterscheidungsmerkmal großer Firmen mehr darstellt und stattdessen ein „Commodity“ geworden ist.[1247] Diese Arbeit zeigt jedoch einen möglichen Grund auf, warum die Diskussion um SOA seit einiger Zeit intensiv geführt wird: SOA beinhaltet das Potential, eine Unternehmens-IT aufzubauen, die über operative Effizienzsteigerungen hinaus auch geschäftliche – also strategische, nachhaltige und integrationsbezogene – Anforderungen unterstützen kann. Eine solche Unternehmens-IT wird einem Unternehmen Wettbewerbsvorteile und Existenzsicherheiten als Nutzen des Einsatzes von IT gewähren. Die Konzepte einer SOA sind einzeln genommen tatsächlich alter Wein. Guter Wein ist eine SOA deswegen, weil diese Konzepte zusammen betrachtet gut geeignet sein können, ein Unternehmen zumindest auf der IT-Seite standhaft, zukunftssicher und wettbewerbsfähig aufzustellen.

[1247] Vgl. Carr /IT/ S. 44-47.

Literaturverzeichnis

Abowd u. a. /Scenario-Based Analysis/
Gregory Abowd, Len Bass, Paul Clements, Rick Kazman: Scenario-Based Analysis of Software Architecture. In: IEEE Software, Vol. 13, Nr. 6, 1996, S. 47 – 55

Adam, McKendrick /Everything in Between/
Colin Adam, Joe McKendrick: From Web Services to SOA and Everything in Between: The Journey Begins. Edinburgh, Doylestown 2005.

Adler /Component Software/
Richard M. Adler: Emerging Standards for Component Software. In: Computer. Vol. 28, Nr. 3, 1995, S. 68 – 77.

Aier, Schönherr /Flexibilisierung/
Stephan Aier, Marten Schönherr: Flexibilisierung von Organisations- und IT-Architekturen durch EAI. In: Stephan Aier, Marten Schönherr (Hrsg.): Enterprise Application Integration. Flexibilisierung komplexer Unternehmensarchitekturen. Berlin 2004, S. 1 – 59.

Akkermans u. a. /Impact/
Henk A. Akkermans, Paul Bogert, Enver Yücesan, Luk N. van Wassenhove: The impact of ERP on supply chain management: Exploratory findings from a European Delphi study. In: European Journal of Operational Research. Vol. 146, Nr. 2, 2003, S. 284 – 301.

Allen /Statement/
Paul Allen: Opening Statement. In: Cutter IT Journal. Vol. 17, Nr. 5, 2004, S. 2 – 4.

Alshawi, Irani, Baldwin /Benchmarking/
Sarmad Alshawi, Zahir Irani, Lynne Baldwin: Benchmarking information technology investment and benefits extraction. In: Benchmarking. An International Journal, Vol. 10, Nr. 4, 2003, S. 414 – 423.

Ambler /Modeling/
Scott W. Ambler: Agile Modeling. Effective Practices for eXtreme Programming and the Unified Process. New York 2002.

Antweiler /Wirtschaftlichkeitsanalyse/
Johannes Antweiler: Wirtschaftlichkeitsanalyse von Informations- und Kommunikationssystemen (IKS). Wirtschaftlichkeitsprofile als Entscheidungsgrundlage. Köln 1995.

Asundi, Kazman, Klein /Economic Approach/
Jayatirtha Asundi, Rick Kazman, Mark Klein: Making Architecture Design Decisions: An Economic Approach. SEI Technical Report CMU/SEI-2002-TR-035, Pittsburgh 2002, http://www.sei.cmu.edu/pub/documents/02.reports/pdf/02tr035-.pdf, Abruf am 2004-09-15

Asundi, Kazman, Klein /Economic Considerations/
Jayatirtha Asundi, Rick Kazman, Mark Klein: Using Economic Considerations to Choose Among Architecture Design Alternatives. SEI Technical Report CMU/SEI-2001-TR-035, Pittsburgh 2001.

Asundi, Kazman, Klein /Quantifying/
Jayatirtha Asundi, Rick Kazman, Mark Klein: Quantifying the Costs and Benefits of Architectural Decisions. In: IEEE: Proceedings of the 23rd International Conference on Software Engineering, Vol. 23, 2001, S. 297 – 306.

Baker /Common Cause/
Colin Baker: Common Cause. In: Airline Business, Vol. 21, Nr. 7, 2005, S. 46 – 47.

Baker /IT Trends/
Colin Baker: IT Trends 2005 Survey. In: Airline Business, Vol. 21, Nr. 7, 2005, S. 36 – 44.

Balzert /Entwicklung/
Helmut Balzert: Die Entwicklung von Software-Systemen. Prinzipien, Methoden, Sprachen, Werkzeuge. Mannheim, Wien, Zürich 1982.

Banke, Krafzig, Slama /Enterprise SOA/
Dirk Krafzig, Karl Banke, Dirk Slama: Enterprise SOA. Service-Oriented Architecture Best Practices. Englewood Cliffs 2004.

Barbacci u. a. /Quality Attributes/
Mario R. Barbacci, Mark H. Klein, Thomas A. Longstaff, Charles B. Weinstock: Quality Attributes. SEI Technical Report CMU/SEI-95-TR-021, Pittsburgh 1995.

Barry /Service-Oriented Enterprise Architecture/
Douglas K. Barry: Getting Ready for a Service Oriented Enterprise Architecture. In: Cutter Executive Report, Vol. 5, Nr. 8, 2002, S. 1 – 24.

Barry /Web Services/
Douglas K. Barry: Web Services and Service-Oriented Architecture: the savvy manager’s guide. Your Road Map to Emerging IT. Amsterdam u. a. 2003.

Bartlett /Building/
Christopher A. Bartlett: Building and managing the transnational: The organizational challenge. In: Michael E. Porter (Hrsg.): Competition in Global Industries. Boston 1986, S. 367-404.

Bass, Clements, Kazman /Software architecture/
Len Bass, Paul Clements, Rick Kazman: Software Architecture in Practice. 2. Auflage, Boston 2003.

Bath, Herr /Post/
Dr. Uwe Bath und Michael Herr: Implementation of a service oriented architecture at Deutsche Post MAIL. In: Stephan Aier, Marten Schönherr (Hrsg.): Enterprise Application Integration – Serviceorientierung und nachhaltige Architekturen. Berlin 2004, S. 279 – 296.

Baumöl, Winter /Qualifikation/
Ulrike Baumöl, Robert Winter: Qualifikation für die Veränderung. In: Hubert Österle, Robert Winter (Hrsg.): Business Engineering. Auf dem Weg zum Unternehmen des Informationszeitalters. 2. Auflage, Berlin, Heidelberg, New York 2003.

Beck /Extreme Programming/
Kent Beck: Extreme Programming explained. Embrace Change. 6. Auflage, Boston u. a. 2001.

Bendzulla, Stülpnagel /Kooperative Entwicklung/
Matthias Bendzulla, Alexander von Stülpnagel: Kooperative Entwicklung einer Informatikplattform in einem großen Verbund. In: Thomas Fischer, Jürgen Moormann (Hrsg.): Handbuch Informationstechnologie in Banken. Wiesbaden 1999, S. 55 – 72.

Berensmann /IT matters/
Dirk Berensmann: IT matters – but who cares? In: Informatik Spektrum. Band 28 Nr. 4, 2005, S. 274 – 277.

Berensmann /Postbank/
Dirk Berensmann: Gesamtbankarchitektur der Deutschen Postbank AG. In: Thomas Fischer, Jürgen Moormann (Hrsg.): Handbuch Informationstechnologie in Banken. 2. Auflage, Wiesbaden 2004, S. 60 – 77.

Boehm /Economics/
Barry W. Boehm: Software Engineering Economics. Englewood Cliffs 1981.

Boehm /Software Engineering/
Barry W. Boehm: Software Engineering. In: IEEE Transactions on Computers, Vol. 25, Nr. 12, 1976, S. 1226-1241.

Boehm /Software-Produktion/
Barry W. Boehm: Wirtschaftliche Software-Produktion. Wiesbaden 1986.

Boehm /Spiral Model/
Barry W. Boehm: A Spiral Model of Software Development and Enhancement. In: IEEE Computer, Vol. 21, Nr. 5, 1988, S. 61 – 72.

Boehm u. a. /Characteristics/
Barry W. Boehm, John R. Brown, Hans Kaspar, Myron Lipow, Gordon J. MacLeod, Michael J. Merrit: Characteristics of Software Quality. Amsterdam, New York, Oxford 1978.

Boehm u. a. /Cocomo II/
Barry W. Boehm, Chris Abts, A. Winsor Brown, Sunita Chulani, Bradford K. Clark, Ellis Horowitz, Ray Madachy, Donald Reifer, Bert Steence: Software Cost Estimation with Cocomo II. Prentice Hall 2000.

Boer /Valuation/
F. Peter Boer: The Valuation of Technology. Business and Financial Issues in R&D. New York u. a. 1999.

Booth, Colomb, Williams /Craft/
Wayne C. Booth, Gregory G. Colomb, Joseph M. Williams: The Craft of Research. 2. Auflage, Chicago, London 2003.

Brandner u. a. /Architecture/
Michael Brandner, Michael Caes, Frank Oellermann, Olaf Zimmermann: Web services-oriented architecture in production in the finance industry. Informatik Spektrum, Vol. 27, Nr. 2, 2004, S. 136 – 145.

Brandt-Pook u. a. /Anwendungsentwicklung/
Hans Brandt-Pook, Bernd Korzen, Joachim Boidol, Hauke Peyen: Anwendungsentwicklung in zeitrestriktiven dynamischen Projekten. In: Wirtschaftsinformatik. Vol 43, Nr. 3, 2001, S. 247 – 254.

Bratthall u. a. /Quality Requirements/
Lars Bratthall, Martin Höst, Enrico Johansson, Andre Wesselén: The importance of quality requirements in software platform development – a survey. In: IEEE Proceedings of the 34th Hawaii International Conference on System Sciences, 2001.

Brynjolfsson /Productivity Paradox/
Erik Brynjolfsson: The Productivity Paradox of Information Technology. In: Communications of the ACM, Vol. 36, Nr. 12, 1993, S. 67 – 77.

Brynjolfsson, Hitt /Paradox Lost/
Erik Brynjolfsson, Lorin Hitt: Paradox Lost? Firm-Level Evidence on the Returns to Information Systems Spending. In: Leslie P. Willcocks, Stephanie Lester (Hrsg.): Beyond the IT productivity paradox. Assessment issues. West Sussex 1999.

Brynjolfsson, Yang /Productivity/
Erik Brynjolfsson, Shinkyu Yang: Information Technology and Productivity: A Review of the Literature. In: Marvin Zelkowitz (Hrsg.): Advances in Computers, Vol. 43, 1996, S. 179-214.

Buhalis /eAirlines/
Dimitrios Buhalis: eAirlines. Strategic and tactical use of ICTs in the airline industry. In: Information & Management, Vol. 41, Nr. 7, 2004, S. 805 – 825.

Burke /Enterprise Architecture/
Brian Burke: Enterprise Architecture or City Planning? Enterprise Planning & Architecture Strategies. MetaGroup Delta, Nr. 2638, Stamford 2003.

CapGemini /IT-Trends 2005/
Cap Gemini: Studie IT-Trends 2005. Paradigmenwechsel in Sicht. http://www.de.capgemini.com/servlet/PB/show/1556864/Capgemini_IT_Trends_2005.pdf, Zugriff 2005-04-04.

Carlson, Tyomkin /Good Design/
Brent Carlson, Dmitry Tyomkin: Service-oriented architecture: Elements of good design. In: Business Integration Journal, Vol. 5, Nr. 1, 2004, S. 13 – 17.

Carr /IT/
Nicholas G. Carr: IT Doesn’t Matter. In: Harvard Business Review. Vol. 81, Nr. 5, 2003, S. 41 – 49.

Carr /Software/
Nicholas G. Carr: Does Software Matter? In: Informatik Spektrum. Band 28, Nr. 4, 2005, S. 271-273.

Carrière, Kazman, Woods /Architectural Quality/
S. Jeromy Carrière, Rick Kazman, Steven G. Woods: Assessing and Maintaining Architectural Quality. In: Proceedings of the Third European Conference on Software Maintenance and Reengineering. Vol. 3, 1999, S. 22 – 30.

Chappell /ESB/
David A. Chappell: Enterprise Service Bus. Sebastopol 2004.

Clements u. a. /Documenting Architecture/
Paul Clements, Felix Bachmann, Len Bass, David Garlan, James Ivers, Reed Little, Robert Nord, Judith Stafford: Documenting Software Architectures. Views and Beyond. Boston u. a. 2003.

Clements, Kazman, Klein /ATAM/
Paul Clements, Rick Kazman, Mark Klein: ATAM: Method for Architecture Evaluation. SEI Technical Report CMU/SEI-2000-TR-004, Pittsburgh 2000.

Clements, Kazman, Klein /Software Architectures/
Rick Kazman, Paul Clements, Mark Klein: Evaluating Software Architectures. Methods and Case Studies. Boston u. a. 2002.

Coleman /Value/
Tom Coleman: IT value for money: going beyond financial analysis. In: Barbara Farbey, David Targett, Frank Land (Hrsg.): Hard Money – Soft Outcomes. Evaluating and Managing the IT Investment. Henley on Thames 1995, S. 59 – 68.

Coombs /Estimation/
Paul Coombs: IT Project Estimation. A Practical Guide to Costing Software. Cambridge 2003.

Crnkovic /Components/
Ivica Crnkovic, Brahim Hnich, Torsten Jonsson, Zeynep Kiziltan: Specification, Implementation, and Deployment of Components. In: Communications of the ACM. Vol. 45, Nr. 10, 2002, S. 35-40.

Cummings, Worley /Organization/
Thomas G. Cummings, Christopher G. Worley: Organization Development and Change. 7. Auflage, Cincinnati 2001.

Cummins /Enterprise Integration/
Fred A. Cummins: Enterprise Integration: An Architecture for Enterprise Application and Systems Integration. New York u. a. 2002

Davenport, Short /Business Process Redesign/
Thomas H. Davenport, James E. Short: The New Industrial Engineering: Information Technology and Business Process Redesign. In: MIT Sloan Management Review, Vol. 31, Nr. 4, 1990, S. 11-27.

Dettling /EAI/
Walter Dettling: EAI oder die Sehnsucht nach einer heilen Welt. In: Andrej Vckovski, Heinrich Meyer, Thomas Brenzikofer, Walter Dettling, Richard Nussdorfer, Thomas Gutzwiller, Sven Pohland (Hrsg.): Netzguide Enterprise Application Integration, Basel 2002, S. 7.

Dobschütz /Wirtschaftlichkeitsanalyse/
Leonard von Dobschütz: Wirtschaftlichkeitsanalyse von Anwendungssystemen: Prämissen und Praxis. In: Information Management. Vol. 7, Nr. 4, 1992, S. 42 – 47.

Dostal, Jeckle /Service-orientierte Architektur/
Wolfgang Dostal, Mario Jeckle: Semantik, Odem einer service-orientierten Architektur. In: Java Spektrum, Vol. 8, Nr. 1, 2004, S. 53-56.

Dostal, Jeckle, Kriechbaum /Semantik/
Wolfgang Dostal, Mario Jeckle, Werner Kriechbaum: Semantik und Web Services: Beschreibung von Semantik. In: Java Spektrum, Vol. 8, Nr. 2, 2004, S. 45-49.

Dumke /Software Engineering/
Reiner Dumke: Software Engineering. Eine Einführung für Informatiker und Ingenieure: Systeme, Erfahrungen, Methoden, Tools. 3. Auflage, Braunschweig, Wiesbaden 2001.

Dustdar, Gall, Hauswirth /Software-Architekturen/
Harald Gall, Schahram Dustar, Manfred Hauswirth: Software-Architekturen für Verteilte Systeme. Berlin u. a. 2003.

Epstein /Drivers/
Marc J. Epstein: The Drivers of Success in Post-Merger Integration. In: Organizational Dynamics, Vol. 33, Nr. 2, 2004, S. 174 – 189.

Erl /Service-Oriented Architecture/
Thomas Erl: Service-Oriented Architecture: A Field Guide to Integrating XML and Web Services. Upper Saddle River 2004.

Farbey, Land, Targett /Taxonomy/
Barbara Farbey, Frank F. Land, David Targett: A taxonomy of information systems applications: the benefits' evaluation ladder. In: European Journal of Information Systems, Vol. 4, Nr. 1, 1995, S. 41 – 55.

Farbey, Targett, Land /Assess/
Barbara Farbey, David Targett, Frank Land: How to assess your IT investment. A study of methods and practice. Oxford u. a. 1993.

Ferstl, Sinz /Wirtschaftsinformatik/
Otto K. Festl, Elmar J. Sinz: Grundlagen der Wirtschaftsinformatik. Band 1. 2. Auflage, München, Wien 1994.

Field /Pain relief/
David Field: Pain relief. In: Airline Business, Vol. 21, Nr. 7, 2005, S. 26 – 27.

Fitzgerald /Evaluating/
Guy Fitzgerald: Evaluating information systems projects: a multidimensional approach. In: Journal of Information Technology. Vol. 13, Nr. 1, 1998, S. 15 – 27.

Frese /Kommunikationseffekte/
Erich Frese: Theorie der Organisationsgestaltung und netzbasierte Kommunikationseffekte. In: Erich Frese, Harald Stöber (Hrsg.): E-Organisation. Wiesbaden 2002, S. 191-241.

Frese /Organisation/
Erich Frese: Grundlagen der Organisation: Konzept – Prinzipien – Strukturen. 8. Auflage, Wiesbaden 2000.

Gaitanides /Prozeßorganisation/
Michael Gaitanides: Prozessorganisation. Entwicklung, Ansätze u. Programme prozeßorientierter Organisationsgestaltung. München 1983.

Gallas /Enterprise Service Integration/
Björn E. Gallas: Enterprise Service Integration (ESI) - Der Weg zu einem service-basierten EAI-Framework unter Einsatz und Erweiterung von Web Services. In: Stephan Aier, Marten Schönherr (Hrsg.): Enterprise Application Integration – Flexibilisierung komplexer Unternehmensarchitekturen. Berlin 2004, S. 176 – 226.

Gallas /Life Cycle/
Björn E. Gallas: Der Aufbau eines Service Life Cycle Managements für eine Service Orientierte Architektur als Brücke zwischen Geschäftsprozess und IT Integration. In: Stephan Aier, Marten Schönherr (Hrsg.): Enterprise Application Integration – Serviceorientierung und nachhaltige Architekturen. Berlin 2004, S. 235 – 255.

Gizanis, Legner, Österle /Kooperative Auftragsabwicklung/
Dimitros Gizanis, Christine Legner, Hubert Österle: Architektur für kooperative Auftragsabwicklung. In: Otto K. Festl, Elmar J. Sinz, Sven Eckbert, Tilman Isselhorst (Hrsg.): Wirtschaftsinformatik 2005. eEconomy, eGovernmanet, eSecurity. Heidelberg 2005, S. 43 – 62.

Gold u. a. /Understanding/
Nicholas Gold, Claire Knight, Andrew Mohan, Malcom Munro: Understanding Service-Oriented Software. In: IEEE Software, Vol. 21, Nr. 2, 2004, S. 71 – 77.

Griffiths, Finlay /Competitive Advantage/
Gareth H. Griffiths, Paul N. Finlay: IS-enabled sustainable competitive advantage in financial services, retailing and manufacturing. In: Journal of Strategic Information Systems. Vol. 13, Nr. 1, 2004, S. 29 – 59.

Gronau /Wandlungsfähige Informationssystemarchitekturen/
Norbert Gronau: Wandlungsfähige Informationssystemarchitekturen – Nachhaltigkeit bei organisatorischem Wandel. Berlin 2003.

Gruhn, Thiel /Komponentenmodelle/
Volker Gruhn, Andreas Thiel: Komponentenmodelle. DCOM, JavaBeans, Enterprise JavaBeans, CORBA. München 2000.

Günther, Tempelmeier /Produktion/
Hans-Otto Günther, Horst Tempelmeier: Produktion und Logistik. 4. Auflage, Berlin u. a. 2000.

Hagen /Credit Suisse/
Claus Hagen: Integrationsarchitektur der Credit Suisse. In: Stephan Aier, Marten Schönherr (Hrsg.): Enterprise Application Integration. Flexibilisierung komplexer Unternehmensarchitekturen. Berlin 2004, S. 62 – 82.

Hahn /Strategische Führung/
Dietger Hahn: Konzepte Strategischer Führung - Entwicklungstendenzen in Theorie und Praxis unter besonderer Berücksichtigung der Globalisierung. In: Dietger Hahn, Bernard Taylor (Hrsg.): Strategische Unternehmungsplanung – Strategische Unternehmungsführung. 8. Auflage, Heidelberg 1999, S. 1037 – 1057.

Hahn, Hungenberg /Controllingkonzepte/
Dietger Hahn, Harald Hungenberg: PuK - Wertorientierte Controllingkonzepte. 6. Auflage, Wiesbaden 2001.

Hammer, Champy /Reengineering/
Michael Hammer, James Champy: Reengineering the Corporation - A Manifesto for Business Revolution. New York 1993.

Hansen /Wirtschaftsinformatik/
Hans Robert Hansen: Wirtschaftsinformatik I. 7. Auflage, Stuttgart 1996.

Hansen, Neumann /Wirtschaftsinformatik/
Hans Robert Hansen, Gustaf Neumann: Wirtschaftsinformatik I. Grundlagen der betrieblichen Informationsverarbeitung. 8. Auflage, Stuttgart 2002.

Hares, Royle /Measuring/
John S. Hares, Duncan Royle: Measuring the value of information technology. Chichester 1994

Harms /Statistik/
Volker Harms: Biomathematik, Statistik und Dokumentation. 7. Auflage, Kiel, Mönkeberg 1998.

Hauser, Löwer /Web Services/
Ulrich M. Löwer, Tobias Hauser: Web Services. Die Standards. Bonn 2004.

Heimann, Kappes /Kostenfalle/
Thomas Heimann, Randolph Kappes: Mit SOA aus der Kostenfalle. In: IT Management, Nr. 11, 2004, S. 44 – 49.

Heinrich, Burgholzer /Systemplanung/
Lutz J. Heinrich, Peter Burgholzer: Systemplanung, Planung und Realisierung von Informations- und Kommunikationssystemen. Band 2, München, Wien 1990.

Herzwurm, Schockert, Mellis /Qualitätssoftware/
Georg Herzwurm, Sixten Schockert, Werner Mellis: Qualitätssoftware durch Kundenorientierung. Die Methode Quality Function Deployment (QFD): Grundlagen, Praxis und SAP R/3 Fallbeispiel. Braunschweig, Wiesbaden 1997.

Herzwurm, Trittmann /Qualität/
Qualität von Internet-Anwendungen: Qualität von Internet-Anwendungen. In: Avcý, Trittmann, Mellis (Hrsg.): Web-Programmierung. Softwareentwicklung mit Internet-Technologien. Grundlagen, Auswahl, Einsatz. XHTML & HTML, CSS, XML, JavaScript, VBScript, PHP, ASP, Java. Wiesbaden 2003. S. 45 – 70.

Hill, Fehlbaum, Ulrich /Organisationslehre/
Wilhelm Hill, Raymond Fehlbaum, Peter Ulrich: Organisationslehre I: Ziele, Instrumente und Bedingungen der Organisation sozialer Systeme. 5. Auflage, Bern, Stuttgart, Wien 1994.

Hoch u. a. /Secrets/
Detlev J. Hoch, Cyriac R. Roeding, Gert Purkert, Sandro K. Lindner, Ralph Müller: Secrets of Software Success. Management Insights from 100 Software Firms around the world. Boston 1999.

Hochstrasser /IT investments/
Beat Hochstrasser: Evaluating IT investments – matching techniques to projects. In: Journal of Information Technology, Vol. 5, Nr. 4, 1990, S. 215 – 221.

Hofmeister, Nord, Soni /Software Architecture/
Christine Hofmeister, Robert Nord, Dilip Soni: Applied Software Architecture. Reading 2000.

Hohmann /Beyond/
Luke Hohmann: Beyond Software Architecture. Creating and Sustaining Winning Solutions. Boston 2003.

Horváth /Controlling/
Péter Horváth: Controlling. 7. Auflage, München 1998.

Horváth /Grundprobleme/
Péter Horváth: Grundprobleme der Wirtschaftlichkeitsanalyse beim Einsatz neuer Informations- und Produktionstechnologien. In: Péter Horváth (Hrsg.): Wirtschaftlichkeit neuer Produktions- und Informationstechnologien. Stuttgart 1988, S. 1 – 14.

Huber /Nachhaltige Entwicklung/
Joseph Huber: Nachhaltige Entwicklung durch Suffizienz, Effizienz und Konsistenz. In: Peter Fritz, Joseph Huber, Hans-Wolfgang Levi (Hrsg.): Nachhaltigkeit in naturwissenschaftlicher und sozialwissenschaftlicher Perspektive. Stuttgart 1995, S. 31 – 46.

Huizen /SOA/
Gordon van Huizen: Get Ready for SOA. In: Business Integration Journal, Vol. 5, Nr. 1, 2004, S. 25 – 27.

Iansiti, Levien /Strategy/
Marco Iansiti, Roy Levien: Strategy as Ecology. In: Harvard Business Review. Vol. 82, Nr. 3, 2004, S. 68 – 78.

IEEE /Architectual Description/
Institute of Electrical and Electronics Engineers (IEEE): IEEE Recommended Practice for Architectural Description of Software Intensive Systems. Std 1471-2000, New York 2003.

IEEE /Glossary/
Institute of Electrical and Electronics Engineers (IEEE): IEEE Standard Glossary of Software Engineering Terminology. Std. 610.12-1990 (R2002), New York 2003.

Inmon, Zachman, Geiger /Data/
William H. Inmon, John A. Zachman, Jonathan G. Geiger: Data Stores, Data Warehousing and the Zachman Framework. Managing Enterprise Knowledge. New York u. a. 1997.

Irani, Love /Frame of Reference/
Zahir Irani, Peter E. D. Love: Developing a frame of reference for ex-ante IT/IS investment evaluation. In: European Journal of Information Systems, Vol. 11, Nr. 1, 2002, S. 74 – 82.

Irani, Love /Propagation/
Zahir Irani, Peter E. D. Love: The propagation of technology management taxonomies for evaluating investments in manufacturing resource planning. Journal of Management Information Systems, Vol. 17, Nr. 3, 2001, S. 161 – 177.

Irani, Love /Systems Evaluation/
Zahir Irani, Peter E. D. Love: Information systems evaluation: past, present and future. In: European Journal of Information Systems. Vol. 10, Nr. 4, 2001, S. 183 – 188.

Irani, Themistocleous, Love /Impact of EAI/
Zahir Irani, Marinos Themistocleous, Peter E.D. Love: The impact of enterprise application integration on information system lifecycles. In: Information & Management. Vol. 41, Nr. 2, 2003, S. 177 – 187.

ISO /9000/
International Organization for Standardization (Hrsg.): Quality Management Systems – Fundamentals and vocabulary. ISO 9000-2000(E). 2. Ausgabe. Genf 2000.

ISO /9126/
International Organization for Standardization (Hrsg.): Software engineering – Product quality – Part 1: Quality model. ISO/IEC 9126-1:2001. Genf 2001.

Jung /IT-Architekturmodell/
Elke Jung: Ein unternehmensweites IT-Architekturmodell als erfolgreiches Bindeglied zwischen der Unternehmensstrategie und dem operativen Bankgeschäft. In: Wirtschaftsinformatik. Vol. 46, Nr. 04, 2004, S. 311 – 322.

Jurkovich /Organisational Environment/
Ray Jurkovich: A Core Typology of Organizational Environments. In: Administrative Science Quarterly. Vol. 19, Nr. 3,1974, S. 380 – 394.

Kaib /EAI/
Michael Kaib: Enterprise Application Integration: Grundlagen, Integrationsprodukte, Anwendungsbeispiele. Wiesbaden 2002 (zugl. Dissertation, Universität Marburg, 2002)

Karch u. a. /SAP/
Steffen Karch, Loren Heilig, Christian Bernhardt, Andreas Hardt, Frank Heidfeld, Roland Pfennig: SAP NetWeaver. Bonn 2004.

Kazman, Bass /Architecture Reviews/
Rick Kazman, Len Bass: Making Architecture Reviews Work in the Real World. In: IEEE Software. Vol. 19, Nr. 1, 2002, S. 67 – 73.

Kazman, Bass /Quality Attributes/
Rick Kazman, Len Bass: Toward Deriving Software Architectures From Quality Attributes. Technical Report CMU/SEI-94-TR-10, Software Engineering Institute, Carnegie Mellon University, Pittsburgh 1994.

Keller /EAI/
Wolfgang Keller: Enterprise Application Integration: Erfahrungen aus der Praxis; Heidelberg 2002.

Kieser, Walgenbach /Organisation/
Alfred Kieser, Peter Walgenbach: Organisation. 4. Auflage, Stuttgart 2003.

Kirchmer /Einführung/
Matthias Kirchmer: Geschäftsprozeßorientierte Einführung von Standardsoftware. Vorgehen zur Realisierung strategischer Ziele. Wiesbaden 1996.

Kirchmer, Scheer /Change Management/
Mathias Kirchmer, August-Wilhelm Scheer: Change Management – der Schlüssel zu Business Process Excellence. In: August-Wilhelm Scheer, Ferri Abolhassan, Wolfram Jost, Mathias Kirchmer (Hrsg.): Change Management im Unternehmen. Prozessveränderungen erfolgreich managen. Berlin u. a. 2003.

Klement /Enterprise Architecture/
Peter Klement: Integrated Enterprise Architecture. Unterschleißheim 2004.

König, Weitzel, Martin /Straight Through Processing/
Tim Weizel, Wolfgang König, Sébastien V. Martin: Straight Through Processing auf XML-Basis im Wertpapiergeschäft. In: Wirtschaftsinformatik. Vol. 45, Nr. 4, 2003, S. 409 – 420.

Krahe /Prozessmanagement/
Andreas Krahe: Unterstützung des Prozeßmanagements mit modernen Informationstechnogien. Wiesbaden 1998.

Krcmar /Bedeutung/
Helmut Krcmar: Bedeutung und Ziele von Informationssystem-Architekturen. In: Wirtschaftsinformatik. Vol. 32, Nr. 5, 1990, S. 395 – 402.

Krebs, Thiel /Sicherheit/
Christian Thiel, Thomas Krebs: Sicherheit in der elektronischen Geschäftsabwicklung. In: Thomas Fischer (Hrsg.), Jürgen Moormann: Handbuch Informationstechnologie in Banken. Wiesbaden 1999, S. 147 – 163.

Krüger /Management/
Wilfried Krüger: Management permanenten Wandels. In: Horst Glaser, Ernst F. Schröder, Axel von Werder (Hrsg.): Organisation im Wandel der Märkte. Wiesbaden 1998, S. 227 – 249.

Krüger, Seelmann-Eggebert /IT-Architektur/
Sascha Krüger, Jörg Seelmann-Eggebert: IT-Architektur-Engineering. Systemkomplexität bewältigen, Kosten senken, Potenziale freisetzten. Bonn 2003.

Krystek /Vertrauen/
Ulrich Krystek: Vertrauen als Basis erfolgreicher strategischer Unternehmungsführung. In: Dietger Hahn, Bernard Taylor (Hrsg.): Strategische Unternehmungsplanung – Strategische Unternehmungsführung. 8. Auflage, Heidelberg 1999, S. 266-288

Laprie /Dependability/
Jean-Claude Laprie: Dependability. Basic Concepts and Terminology in English, French, German, Italian and Japanese. In: Algirdas Avizienis, Hermann Kopetz, Jean-Claude Laprie (Hrsg.): Dependable Computing and Fault-Tolerant Systems. Vol. 5, Wien, New York 1992.

Laudon, Laudon /Business/
Kenneth C. Laudon, Jane Price Laudon: Business Information Systems. A Problem-Solving Approach. Fort Worth u. a. 1991.

Lederer /IT-Gesamtbankarchitektur/
Anno Lederer: IT-Gesamtbankarchitektur in der Genossenschaftsorganisation. In: Thomas Fischer (Hrsg.), Jürgen Moormann: Handbuch Informationstechnologie in Banken. 2. Auflage, Wiesbaden 2004, S. 79 – 93.

Lewin, Hunter /Information Technology/
Arie Y. Lewin, Starling D. Hunter: Information Technology & Organizational Design: A Longitudinal Study of Informtaion Technology Implementations in the U.S. Retailing Industry, 1980-1996. In: Horst Glaser, Ernst F. Schröder, Axel von Werder (Hrsg.): Organisation im Wandel der Märkte. Wiesbaden 1998, S. 251 – 286.

Ließmann, Wetzke /Integrationsinfrastruktur/
Harald Ließmann, Jan Wetzke: Entwicklung und Management einer Service-orientierten Integrationsinfrastruktur mithilfe von BPEL und BPMN. In: Stephan Aier, Marten Schönherr (Hrsg.): Enterprise Application Integration – Serviceorientierung und nachhaltige Architekturen. Berlin 2004, S. 198 – 228.

Linß /Nutzeffekte/
Heinz Linß: Integrationsabhängige Nutzeffekte der Informationsverarbeitung. Vorgehensmodell und empirische Ergebnisse. Dissertation, Wiesbaden 1995.

Linthicum /Application Integration/
David S. Linthicum: Enterprise Application Integration. Boston u. a. 2000.

Linthicum /B2B/
David S. Linthicum: B2B Application Integration: e-Business-Enable Your Enterprise. Boston 2001.

Linthicum /Next Generation/
David S. Linthicum: Next Generation Application Integration. From Simple Information to Web Services. Boston u. a. 2004.

Linthicum /SOA/
David S. Linthicum: Leveraging SOA & Legacy Systems. In: Business Integration Journal. Vol. 5, Nr. 7, 2004, S. 14 – 17.

Lublinsky, Tyomkin /Dissecting SOA/
Boris Lublinsky, Dmitry Tyomkin: Dissecting service-oriented architectures. In: Business Integration Journal, Vol. 4, Nr. 10, 2003, S. 52 – 58.

Lucas /Information Technology/
Henry C. Lucas Jr.: Information Technology and the Productivity Paradox. New York, Oxford 1999.

MacCormack, Verganti, Iansiti /Products/
Alan MacCormack, Roberto Verganti, Marco Iansiti: Developing Products on Internet Time: The Anatomy of a Flexible Development Process. In: Management Science. Vol. 47, Nr. 1, 2001, S. 133 – 150.

Macharzina /Unternehmensführung/
Klaus Macharzina: Unternehmensführung. Das internationale Managementwissen. 3. Auflage, Wiesbaden 1999.

McCall /Quality factors/
James A. McCall: Quality factors. In: John L. Marciniak (Hrsg.): Encyclopedia of Software Engineering. New York, 1994, S. 958 – 969.

McCoy, Natis /Mainstream/
David McCoy, Yefim Natis: Service-Oriented Architecture: Mainstream Straight Ahead. Gartner Research LE-19-7652, 2003.

McGovern u. a. /Architecture/
James McGovern, Sameer Tyagi, Michael Stevens, Sunil Mathew: Java Web Services Architecture. San Francisco 2003.

Mellis /Lehre/
Werner Mellis: Lehre als Evaluationsobjekt. Einführung und Grundlegung. In: Lutz J. Heinrich, Irene Häntschel (Hrsg.): Evaluation und Evaluationsforschung in der Wirtschaftsinformatik. Handbuch für Praxis, Lehre und Forschung. München, Wien 2000, S. 383 – 394.

Mellis /Process/
Werner Mellis: Process and Product Orientation in Software Development and their Effect on Software Quality Management. In: Martin J. Wieczorek, Dirk B. Meyerhoff (Hrsg.): Software Quality – State of the Art in Management, Testing, and Tools. Berlin 2000, S. 3 – 15.

Mellis /Projektmanagement/
Werner Mellis: Projektmanagement der SW-Entwicklung. Eine umfassende und fundierte Einführung. Wiesbaden 2004.

Mellis /Turbulent Times/
Mellis, Werner: Software Quality Management in Turbulent Times - Are there Alternatives to Process oriented Software Quality Management? In: Software Quality Journal. Vol. 7, Nr. 3, 1998, S. 277-295.

Mellis u. a. /Software/
Werner Mellis, Georg Herzwurm, Uwe Müller, Harald Schlang, Sixten Schockert, Ralph Trittmann: Rapid Software Development. Aachen 2001.

Mertens /Gefahren/
Peter Mertens: Gefahren für die Wirtschaftsinformatik. Risikoanalyse eines Faches. In: Otto K. Festl, Elmar J. Sinz, Sven Eckbert, Tilman Isselhorst (Hrsg.): Wirtschaftsinformatik 2005. eEconomy, eGovernmanet, eSecurity. Heidelberg 2005, S. 1733 – 1754.

META Group /Approaches/
META Group (Hrsg.): Practical Approaches to Service-Oriented Architecture: Meeting the Demand Today and Tomorrow. Meta Group White Paper 2003.

Mingers /Critical Realism/
John Mingers: Re-establishing the Real: Critical Realism and Information Systems. In: John Mingers, Leslie P. Willcocks (Hrsg.): Social Theory and Philosophy for Information Systems, Wiley, 2004, S. 372 – 407.

Mingers /Real-izing information systems/
John Mingers: Real-izing information systems: critical realism as an underpinning philosophy for information systems. In: Information and Organization, Vol. 14, Nr. 2, 2004, S. 87 – 103.

Mintzberg /Structuring/
Henry Mintzberg: Structuring of Organizations. A Synthesis of the Research. Englewood Cliffs 1979.

Moormann /Bankinformatik/
Jürgen Moormann: Umbruch der Bankinformatik. Status quo und Perspektiven für eine Neugestaltung. In: Thomas Fischer (Hrsg.), Jürgen Moormann: Handbuch Informationstechnologie in Banken. Wiesbaden 1999, S. 3-20.

Moormann /Umbruch/
Jürgen Moormann: Umbruch in der Bankinformatik - Status quo und Perspektiven für eine Neugestaltung. In: Jürgen Moormann, Thomas Fischer (Hrsg.): Handbuch Informationstechnologie in Banken. Wiesbaden 1999, S. 4 – 20.

Morgan /Bilder/
Gareth Morgan: Bilder der Organisation. Stuttgart, 1997

Moro, Lehner /Reengineering/
Martin Moro, Franz Lehner: Reengineering und Entwicklung einer neuen Softwarearchitektur am Beispiel Heidelberg Eye Explorer. In: Peter Kleinschmidt, Franz Lehner (Hrsg.): Schriftenreihe Wirtschaftsinformatik. Diskussionsbeitrag W-05-04.

Müller /Software-Engineering/
Erwin Müller: Software-Engineering als kreativer Prozess. Meta-Kriterien für die Entwicklung von Informatiklösungen. Bern u. a. 1988.

Murphy, Simon /Intangible Benefits/
Kenneth E. Murphy, Steven J. Simon: Intangible benefits valuation in ERP projects. In: Information Systems Journal, Vol. 12, Nr. 4, 2002, S. 301 – 320.

Nadhan /Service-Oriented Evolution/
Easwaran G. Nadhan: Seven Steps to a Service-Oriented Evolution. In: Business Integration Journal, Vol. 5, Nr. 1, 2004, S. 41 – 44.

Nagel /Nutzen/
Kurt Nagel: Nutzen der Informationsverarbeitung. Methoden zur Bewertung von strategischen Wettbewerbsvorteilen, Produktivitätsverbesserungen und Kosteneinsparungen. 2. Aufl., München, Wien 1990.

Natis /SOA Scenario/
Yefim Natis: Service-Oriented Architecture Scenario. Gartner Research AV-19-6751, 2003.

Natis /SOA/
Yefim Natis: Service-oriented Architecture (SOA) ushers in the next era in business software engineering. In: Business Integration Journal, Vol. 5, Nr. 5, 2004, S. 23 – 25.

Natis, Schulte /Introduction/
Yefim Natis, Roy W. Schulte: Introduction to service-oriented architecture. Gartner Research SPA-19-5971, 2003.

Naur /Computing/
Peter Naur: Computing. A Human Activity. Reading u. a. 1992.

Oestereich u. a. /Objektorientierung/
Bernd Oestereich (Hrsg.), Peter Hruschka, Nicolai Josuttis, Hartmut Kocher, Hartmut Krasemann, Markus Reinhold: Erfolgreich mit Objektorientierung. Vorgehensmodelle und Managementpraktiken für die objektorientierte Softwareentwicklung. 2. Auflage, Wien, München, Oldenburg 2001.

Oey u. a. /Einführende Darstellung/
Kai J. Oey, Holger Wagner, Simon Rehbach, Andrea Bachmann: Mehr als alter Wein in neuen Schläuchen. Eine einführende Darstellung des Konzepts der serviceorientierten Architekturen. In: Aier, Schönherr (Hrsg.): Unternehmensarchitekturen und Systemintegration. Reihe Enterprise Architecture, Band 3, Berlin 2005, S. 197 – 215.

Österle /Integration/
Hubert Österle: Integration: Schlüssel zur Informationsgesellschaft. In: Hubert Österle, Rainer Riehm, Petra Vogler (Hrsg.): Middleware - Grundlagen, Produkte und Anwendungsbeispiele für die Integration heterogener Welten. Braunschweig, Wiesbaden 1996, S. 1 – 23.

Österle, Fleisch, Felt /Business Networking/
Hubert Österle, Elgar Fleisch, Rainer Alt: Business Networking in der Praxis. Berlin, Heidelberg, New York 2002.

Oxman, Smith /Structual Change/
Jeffrey A. Oxman, Brian D. Smith: The Limits of Structual Change. In: MIT Sloan Management Review. Vol. 45, Nr. 1, S. 77 – 82.

Pasley /BPEL/
James Pasley: How BPEL and SOA Are Changing Web Services Development. In: IEEE Internet Computing. Vol. 9, Nr. 3, 2005, S. 60 – 67.

Pfau /Informationsmanagement/
Wolfgang Pfau: Betriebliches Informationsmanagement. Flexibilisierung der Informationsinfrastruktur. In: Picot, Reichwald (Hrsg.): Reihe: Markt und Unternehmensentwicklung. Wiesbaden 1997.

Picot, Reichwald, Wiegand /Grenzenlose Unternehmung/
Arnold Picot, Ralf Reichwald, Rolf T. Wiegand: Die Grenzenlose Unternehmung – Information, Organisation und Management. 4. Auflage, Wiesbaden 2001.

Porter /Advantage/
Michael E. Porter: Competitive Advantage. Creating and Sustaining Superior Performance. 1. Export Ausgabe, New York u. a. 2004.

Porter /Internet/
Michael E. Porter: Strategy and the Internet. In: Harvard Business Review. Vol. 79, Nr. 3, 2001, S. 62 – 78.

Porter /Strategy/
Michael E. Porter: Competitive Strategy. Techniques for Analyzing Industries and Competitors. 1. Export Ausgabe, New York u. a. 2004.

Posch, Birken, Gerdom /Basiswissen/
Torsten Posch, Klaus Birken, Michael Gerdom: Basiswissen Softwarearchitektur. Verstehen, entwerfen, bewerten und dokumentieren. Heidelberg 2004.

Potthof /Empirische Studien/
Ingo Potthof: Empirische Studien zum wirtschaftlichen Erfolg der Informationsverarbeitung. In: Wirtschaftsinformatik. Vol. 40, Nr. 1, 1998, S. 54 – 65.

Raasch /Systementwicklung/
Jörg Raasch: Systementwicklung mit Strukturierten Methoden. Ein Leitfaden für Praxis und Studium.3. Auflage, München, Wien 1993.

Reifer /Business Case/
Donald J. Reifer: Making the Software Business Case. Improvement by the Numbers. Boston 2001.

Reinertsen /System Architecture/
Donald G. Reinertsen: Turning System Architecture into Profits. In: Cutter IT Journal. Vol. 17, Nr. 1, 2004, S. 20 – 25.

Reinertsen, Smith /Half The Time/
Preston Smith, Donald G. Reinertsen: Developing Products in Half the Time. 2. Ausgabe, New York 1998.

Renkema, Berghout /Methodologies/
Theo J. W. Renkema, Egon W. Berghout: Methodologies for information systems investment evaluation at the proposal stage: a comparative review. In: Information and Software Technology. Vol. 39, Nr. 1, 1997, S. 1 – 13.

Rine, Nada, Jaber /Adapters/
Rine, Nada, Jaber: Using Adapters to Reduce Interaction Complexity in Reusable Component-Based Software Development. In: Proceedings of the 1999 symposium on Software reusability. Vol. 3, 1999, S. 37 – 43.

Rood /Enterprise Architecture/
Melody A. Rood: Enterprise architecture: definition, content, and utility. In: IEEE Proceedings of the Third Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, Vol. 3, 1994, S. 106 – 111.

Rothman /Architectual Infrastructure/
Johanna Rothmann: Investing in Architectural Infrastructure: A Business Conversation. In: Cutter IT Journal. Vol. 17, Nr. 1, 2004, S. 12 – 19.

Ruh, Maginnis, Brown /EAI/
William A. Ruh, Francis X. Maginnis, William J. Brown: Enterprise Application Integration: A Wiley Tech Brief. New York u. a. 2001.

Rumbaugh u. a. /Modellieren/
James Rumbaugh, Michael Blaha, William Premerlani, Frederick Eddy, William Lorensen: Objektorientiertes Modellieren und Entwerfen. München, Wien, London 1993.

Rupietta /Benutzerdokumentation/
Walter Rupietta: Benutzerdokumentation für Softwareprodukte. Mannheim, Wien, Zürich 1987.

Sassone /CBA of IS/
Peter G. Sassone: Cost Benefit Analysis of Information Systems: A Survey of Methodologies. In: Proceedings of the ACM Conference on Supporting Group Work, New York 1988, S. 126 – 133.

Sassone, Schaffer /CBA/
Peter G. Sassone, William A. Schaffer: Cost-Benefit Analysis. A Handbook. London 1978

Scheckenbach /Geschäftsprozessintegration/
Rainer Scheckenbach: Semantische Geschäftsprozessintegration. Dissertation, Wiesbaden 1997.

Schlamann /Service Orientation/
Hermann Schlamann: Service Orientation: An Evolutionary Approach. In: Cutter IT Journal, Vol. 17, Nr. 5, 2004, S. 5-13.

Schmietendorf, Dimitrov, Dumke /JavaBeans/
Andreas Schmietendorf, Evgeni Dimitrov, Reiner Dumke: Enterprise JavaBeans. Bonn 2002.

Schreyögg /Organisation/
Georg Schreyögg: Organisation. Grundlagen moderner Organisationsgestaltung. 3. Auflage, Wiesbaden 1999.

Schulze /Computer/
Hans Herbert Schulze: Computer Enzyklopädie. Reinbeck 1989.

Schumann /Nutzeffekte/
Matthias Schumann: Betriebliche Nutzeffekte und Strategiebeiträge der großintegrierten Informationsverarbeitung. Berlin u. a. 1992.

Schumann /Wirtschaftlichkeitsbeurteilung/
Matthias Schumann: Wirtschaftlichkeitsbeurteilung für IV-Systeme. In: Wirtschaftsinformatik. Vol. 35, Nr. 2, 1993, S. 167 – 178.

Schuppe, Cassens /IT im Handelsraum/
Heiko Cassens, Karen Schuppe: Informationstechnologie im Handelsraum. In: Thomas Fischer, Jürgen Moormann (Hrsg.): Handbuch Informationstechnologie in Banken. Wiesbaden 2004, S. 399 – 412.

SEI /Definitions/
Software Engineering Institute (SEI) (Hrsg.): How Do You Define Software Architecture? http://www.sei.cmu.edu/architecture/definitions.html, Zugriff 2005-01-01.

Seibt /Begriffe und Aufgaben/
Dietrich Seibt: Begriffe und Aufgaben des Informationsmanagements – ein Überblick. In: Dieter B. Preßmar (Hrsg.): Informationsmanagement. Wiesbaden 1993, S. 3 – 30.

Seibt /Probleme und Aufgaben/
Dietrich Seibt: Ausgewählte Probleme und Aufgaben der Wirtschaftsinformatik. In: Wirtschaftsinformatik, Band 32, Nr. 1, 1990, S. 7 – 19.

Seibt /Vorgehensmodell/
Dietrich Seibt: Vorgehensmodell. In: Peter Mertens, Andrea Back, Jörg Becker, Wolfgang König, Hermann Krallmann, Bodo Rieger, August-Wilhelm Scheer, Dietrich Seibt, Peter Stahlknecht, Horst Strunz, Rainer Thome, Hartmut Wedekind (Hrsg.): Lexikon der Wirtschaftsinformatik. 3. Aufl., Berlin, Heidelberg, New York 1997, S. 431-434.

Shin /Strategic Choice/
Namchul Shin: The impact of information technology on financial performance: the importance of strategic choice. In: European Journal of Information Systems, Vol. 10, Nr. 4, 2001, S. 227 – 236.

Sholler /Case/
Daniel Sholler: Service-Oriented Architectures: Part 3 – The Case for Services. In: MetaGroup Delta. Nr. 1246, 2003.

Sholler /Governance/
Daniel Sholler: Service-Oriented Architectures: Part 2 – Governance for Enterprise Services. In: MetaGroup Delta. Nr. 1245, 2003.

Sholler /Resuable Enterprise Services/
Daniel Sholler: Service-Oriented Architectures: Part 1 – Defining Resuable Enterprise Services. In: Meta Group Delta, Nr. 1244, 2003.

Sinz /Architektur/
Elmar J. Sinz: Architektur von Informationssystemen. In: Peter Rechenberg, Gustav Pomberger (Hrsg.): Informatik-Handbuch. München 1997, S. 875 – 887.

Smithson, Hirschheim /Old Problem/
Steve Smithson and Rudy Hirschheim: Analysing information systems evaluation: another look at an old problem. In: European Journal of Information Systems. Vol. 7, Nr. 3, 1998, S. 158 – 174.

Solingen /ROI/
Rini van Solingen: Measuring the ROI of Software Process Improvement. In: IEEE Software. Vol. 21, Nr. 3, 2004, S. 32 – 38.

Sommerville /Software Engineering/
Ian Sommerville: Software Engineering. 6. Auflage, München 2001.

Stahlknecht, Hasenkamp /Wirtschaftsinformatik/
Peter Stahlknecht, Ulrich Hasenkamp: Einführung in die Wirtschaftsinformatik. 9. Auflage. Berlin, Heidelberg 1999.

Stelzer /Möglichkeiten/
Dirk Stelzer: Möglichkeiten und Grenzen des prozeßorientierten Software-Qualitätsmanagements. Habilitationsschrift. Köln 1998.

Stiemerling /Web-Services/
Oliver Stiemerling: Web-Services als Basis für evolvierbare Softwaresysteme. In: Wirtschaftsinformatik. Vol. 44, Nr. 5, 2002, S. 435–445.

Strabel /Neue Architektur/
Peter Strabel: Neue Architektur für die spartenneutrale Kontoführung. In: Thomas Fischer, Jürgen Moormann (Hrsg.): Handbuch Informationstechnologie in Banken. Wiesbaden 1999, S. 87 – 102.

Strunz /Anwendungsarchitektur/
Horst Strunz: Anwendungsarchitektur. In: Peter Mertens, Andrea Back, Jörg Becker, Wolfgang König, Hermann Krallmann, Bodo Rieger, August-Wilhelm Scheer, Dietrich Seibt, Peter Stahlknecht, Horst Strunz, Rainer Thome, Hartmut Wedekind (Hrsg.): Lexikon der Wirtschaftsinformatik. 3. Auflage, Berlin, Heidelberg, New York 1997, S. 35 – 37.

Svahnberg, Wohlin /Quality Attributes/
Mikael Svahnberg, Claes Wohlin: An Investigation of a Method for Identifying a Software Architecture Candidate with Respect to Quality Attributes. In: Empirical Software Engineering, Vol. 10, Nr. 2, 2005, S. 149 – 181.

Szyperski /Strategisches Informationsmanagement/
Norbert Szyperski: Strategisches Informationsmanagement im technologischen Wandel. Fragen zur Planung und Implementierung von Informations- und Kommunikationssystemen. BIFOA Arbeitspapier 79AP22, Köln 1979.

Tannhäuser, Umek /Architekturmanagement/
Carsten Tannhäuser, Alexander Umek: Wert schaffen durch nutzenorientiertes Architekturmanagement in EAI Projekten. In: Stephan Aier, Marten Schönherr (Hrsg.): Enterprise Application Integration – Serviceorientierung und nachhaltige Architekturen. Berlin 2004, S. 53 – 70.

Teubner, Rentmeister, Klein /IT-Fitness/
Alexander Teubner, Jahn Rentmeister, Stefan Klein: IT-Fitness für das 21. Jahrhundert. Konzeption eines Evaluationsinstruments. In: Lutz J. Heinrich, Irene Häntschel (Hrsg.): Evaluation und Evaluationsforschung in der Wirtschaftsinformatik. Handbuch für Praxis, Lehre und Forschung. München, Wien 2000, S. 75 – 92.

Themistocleous, Irani /Benchmarking/
Marinos Themistocleous, Zahir Irani: Benchmarking the benefits and barriers of application integration. In: Benchmarking. An International Journal, Vol. 8, Nr. 4, 2001 S. 317 – 331.

Thorp /Information Paradox/
John Thorp: The Information Paradox. Realizing the Business Benefits of Information Technology. Toronto 1998.

Trapp, Otto /Einsatzmöglichkeiten/
Ralph C. Trapp, Alfred Otto: Einsatzmöglichkeiten von EAI bei Mergers & acquisitions. In: HMD – Praxis der Wirtschaftsinformatik. Nr. 225, 2002, S. 102 – 113.

Trittmann u. a. /Sieg der Moderne/
Ralph Trittmann, Werner Mellis, Holger Wagner, Rasmus Bergmann, Oral Avci: Sieg der Moderne über die Tradition? Ergebnisse einer empirischen Untersuchung zur Projektgestaltung in der Softwareentwicklung. In: Projektmanagement aktuell. Nr. 5, Vol. 15, 2005, S. 10-15.

Vogt /Betriebssysteme/
Carsten Voigt: Betriebssysteme. Heidelberg u. a. 2001.

W3C /Glossary/
W3C (Hrsg.): Web Services Glossary. W3C Working Group Note 11th February 2004. http://www.w3.org/TR/2004/NOTE-ws-gloss-20040211, Zugriff 2005-01-01.

Walter, Spitta /Evaluation/
Sascha G. Walter, Thorsten Spitta: Approaches to the Ex-ante Evaluation of Investments into Information Systems. In: Wirtschaftsinformatik. Vol. 46, Nr. 3, 2004, S. 171-180.

Wang u. a. /Integrated Quality of Service/
Guijin Wang, Alice Chen, Changzhou Wang, Casey Fung, Stephen Uczekaj: Integrated Quality of Service (QoS) Management in Service-Oriented Enterprise Architectures. In: Proceedings of the 8th IEEE International Enterprise Distributed Object Computing Conference (EDOC), Vol. 8, Montenery 2004, S. 20 – 24.

Wehle /Business Continuity/
Heinz-Jürgen Wehle: Business Continuity in der IT-Produktion. In: Thomas Fischer, Jürgen Moormann (Hrsg.): Handbuch Informationstechnologie in Banken. Wiesbaden 2004, S. 219 – 235.

Weltz, Ortmann /Softwareprojekt/
Friedrich Weltz, Rolf G. Ortmann: Das Softwareprojekt: Projektmanagement in der Praxis. Frankfurt u. a. 1992.

Werder, Grundei, Talaulicar /Unternehmenskommunikation/
Axel v. Werder, Jens Grundei, Till Talaulicar /Unternehmenskommunikation. In: Erich Frese, Harald Stöber (Hrsg.): E-Organisation. Wiesbaden 2002, S. 395 – 423.

Wermke, Kunkel-Razum, Scholze-Stubenrecht /Duden/
Matthias Wermke, Kathrin Kunkel-Razum, Werner Scholze-Stubenrecht (Hrsg.): Duden. Die deutsche Rechtschreibung. Band 1. 23. Auflage, 2004.

Willgosch /Informationslieferanten/
Volker Willgosch: IT-Konzepte der Informationslieferanten für Banken. In: Thomas Fischer, Jürgen Moormann (Hrsg.): Handbuch Informationstechnologie in Banken. Wiesbaden 1999, S. 300 – 311.

Winter /Modell/
Robert Winter: Ein Modell zur Visualisierung der Anwendungslandschaft als Grundlage der Informationssystem-Architekturplanung. In: Joachim Schelp, Robert Winter (Hrsg.): Integrationsmanagement. Planung, Bewertung und Steuerung von Applikationslandschaften. Berlin 2006, S. 1-30.

Wolfe, Putler /Stakeholder Groups/
Richard A. Wolfe, Daniel S. Putler: How Tight Are the Ties that Bind Stakeholder Groups? In: Organization Science. Vol. 13, Nr. 1, 2002, S. 64 – 80.

Woods /Enterprise Services Architecture/
Dan Woods: Enterprise Services Architecture. Bonn 2004.

Zachman /Cost-Justify Architecture/
John A. Zachman: You Can’t Cost-Justify Architecture. In: Business Rules Journal, Vol. 29, Nr. 3, 2001, http://www.brcommunity.com/b059.php, Zugriff 2004-09-25.

Zachman /Framework/
John A. Zachman: A framework for information systems architecture. In: IBM Systems Journal, Vol. 38, Nr. 2-3, 1999, S. 454 – 470.

Zahavi /Integration/
Ron Zahavi: Enterprise Application Integration with CORBA. Component and Web-Based Solutions. New York u. a. 2000.

Zuser, Grechenig, Köhle /Software Engineering/
Wolfgang Zuser, Thomas Grechenig, Monika Köhle: Software Engineering mit UML und dem Unified Process. München 2004.

Danksagung

Die vorliegende Arbeit entstand während und nach meiner Tätigkeit am Lehrstuhl für Wirtschaftsinformatik, Systementwicklung, an der Universität zu Köln.

Mein ganz besonderer Dank gilt meinem Doktorvater Herrn Prof. Dr. Werner Mellis für die wohlwollende Unterstützung dieser Arbeit und für die wertvollen Erfahrungen, die ich als Schüler und Mitarbeiter an seinem Lehrstuhl sammeln konnte.

Herrn Prof. Dr. Dietrich Seibt danke ich für sein Interesse an der Arbeit und die Übernahme des Zweitgutachtens.

Darüber hinaus gilt mein Dank den Teilnehmern der Studie, die mir interessante Einblicke in die praktische Arbeit ermöglichten, allen Mitarbeitern des Lehrstuhls, auch den studentischen Hilfskräften, die durch ihr kollegiales Verhalten und tatkräftige Hilfsbereitschaft zum Gelingen dieser Arbeit beigetragen haben. Erwähnen möchte ich auch die Diplomanden Andrea Bachmann, Simon Rehbach und Jens Becker sowie meinen Kollegen Holger Wagner für ihr Interesse an dem Thema und den daraus resultierenden Anregungen für meine Arbeit. Insbesondere danke ich unserer Institutssekretärin Brigitte Bernd für ihre stetige Hilfsbereitschaft und meinem Freund und Kollegen Frank Trefflich für viele Stunden wertvoller Diskussion.

Meinen Freunden Stefanie Ohnrich, Christiane Broich und Simone Büttner danke ich für die Durchsicht und meiner Lebensgefährtin Anke Lehmacher für ihre Nachsicht und Unterstützung während der Erstellung der Arbeit. Mailin und Erik, meinen Geschwistern, und all den Freunden, die ich in den vergangenen Monaten getroffen und gesprochen habe, danke ich für die gekonnte Ablenkung, z. B. durch Fliegen und Doppelkopf, die meinen Kopf freigemacht haben und mir die Möglichkeit gaben, am nächsten Tag frisch und gut gelaunt weiterzumachen.

Besonders herzlich danke ich meinen Eltern für ihr unerschütterliches Vertrauen in mich und die unschätzbare Unterstützung, welche ich mein ganzes Leben lang erhalten habe.

Lebenslauf

Geburt	Tübingen, 08. Mai 1976
Familienstand	Ledig
Eltern	Brigitte und Dr. rer. nat. Jan Oey Midlothian, Virginia, USA

Ausbildung

10/1996 – 09/2002	Universität zu Köln **Diplom** (Prädikatsexamen) im Fach Wirtschaftsinformatik
10/1999 – 02/2000	Warsaw School of Economics Auslandssemester, CEMS-Programm (Community of European Management Schools), Warschau, Polen

Berufserfahrung und Praktika

11/2005 – dato:	Accenture GmbH Unternehmensberater, Strategic IT Effectiveness
11/2002 – 10/2005:	Universität zu Köln, Lehrstuhl für Wirtschaftsinformatik, Systementwicklung Doktorand, geschäftsführender Assistent, Mitglied im Prüfungsausschuss für den Diplomstudiengang Wirtschaftsinformatik, Stellvertretendes Mitglied der Engeren Fakultät der Wirtschafts- und Sozialwissenschaftlichen Fakultät
01/1999 – dato:	Synergetix, Köln/Hannover, Gründer und Eigner
10/1998 – 03/2002:	Universität zu Köln, Lehrstuhl für Wirtschaftsinformatik, Systementwicklung, Studentische Hilfskraft
08/2000 – 09/2000:	BMW AG, München, Praktikant
10/1996 – 10/1998:	Philip Morris Research Laboratories GmbH, Köln, Werkstudent

Wissenschaftliches Engagement

Arbeitskreis „Serviceorientierte Architekturen"

Stellvertretender Sprecher des Arbeitskreises „Serviceorientierte Architekturen (SOA)“ der Gesellschaft für Informatik. Inhaltli-

che Gestaltung, Organisation und Moderation der im 6-Wochen-Ryhtmus stattfindenden Meetings.

Workshop „Serviceorientierte Architekturen - Zusammenwirken von Business & IT“

auf der Jahrestagung der Gesellschaft für Informatik, der Informatik2005 am 22.09.2005, Bonn. Organisationsleitung, Auswahl von Vorträgen und Koordination des Programmkomitees.