February 2005

Entwicklung von Zielen und Messgrößen zur Steuerung der Applikationsintegration

Alexander Schwinn
Universität St. Gallen

Robert Winter
Universität St. Gallen

In: Ferstl, Otto K, u.a. (Hg) 2005. *Wirtschaftsinformatik 2005: eEconomy, eGovernment, eSociety*; 7. Internationale Tagung Wirtschaftsinformatik 2005. Heidelberg: Physica-Verlag

ISBN: 3-7908-1574-8

Entwicklung von Zielen und Messgrößen zur Steuerung der Applikationsintegration

Alexander Schwinn, Robert Winter
Universität St. Gallen

Zusammenfassung: Die Effektivität und Effizienz des Informationssystems der Unternehmung werden wesentlich vom Integrationsgrad der Applikationen beeinflusst. Um den Integrationsgrad systematisch planen und steuern zu können, muss ein entsprechendes Zielsystem spezifiziert werden. Dazu werden in diesem Beitrag fünf Ziele der Applikationsintegration identifiziert. Zu den einzelnen Zielen werden jeweils Kennzahlen diskutiert, die eine Messung der Zielerreichung ermöglichen. Zudem werden Abhängigkeiten und Wechselwirkungen zwischen einzelnen Zielen qualitativ untersucht und daraus Hypothesen abgeleitet.

Schlüsselworte: Applikationsintegration, Architekturmanagement, Enterprise Applikation Integration, EAI, Applikationsarchitektur

1 Einführung

Die Planung und Steuerung der Informationssystem-Architektur wird gleichermaßen von der Forschung (z.B. im deutschsprachigen Raum [Krcm90], [Öst+92]) wie auch von Beratungs- bzw. Softwareunternehmen adressiert [Zach87]. Architekturmodelle dienen dazu, einen ganzheitlichen Überblick über das Informationssystem der Unternehmung zu schaffen und Gestaltungsentscheidungen zu unterstützen [Krcm03, S. 39ff.].

Seit einigen Jahren wird die Planung und Steuerung der Informationssystem-Architektur zunehmend mit entsprechenden Ansätzen auf anderen Betrachtungsebenen (z.B. „Business Architecture" [McDa99], [You+99]) integriert und als umfassendes Gestaltungskonzept „Unternehmensarchitektur" („Enterprise Architecture", vgl. [Malh96], [Zach99] und [MaRo00]) diskutiert. Mit ISO 15704 Annex A, ISO/CEN FDIS 19439 sowie ISO/IEC 15288 liegen zudem erste Normen vor, die die Gestaltung der Informationssystem-Architektur als Teil eines umfassenderen Gestaltungsansatzes beschreiben.

Aufgrund der Vielgestaltigkeit der Komponenten des Informationssystems und der Möglichkeit, dessen Struktur sowohl aus fachlicher wie auch aus technischer Perspektive zu beschreiben, sind unterschiedliche Architekturmodelle für das Informationssystem gebräuchlich. In diesem Beitrag wird die Applikationsarchitektur

betrachtet, weil sie (1) ein Modell aus fachlicher Sicht darstellt und weil (2) die Gestaltung von Applikationen und Schnittstellen zwischen Applikationen wesentlichen Einfluss auf Integrationskosten hat. Die Applikationsarchitektur beschreibt Applikationen und ihr Zusammenwirken [Wint03, S. 94]. Ein wichtiges Dokument im Zusammenhang mit der Analyse und Gestaltung der Applikationsarchitektur ist die sog. Applikationslandschaft, d.h. die aggregierte graphische Darstellung der wichtigsten Applikationen und meist auch der wichtigsten Schnittstellen zwischen Applikationen. Das Ziel der Applikationsgestaltung, zu dessen Unterstützung die Planung und Steuerung der Applikationsarchitektur beitragen soll, ist die optimale Integration von Applikationen.

Ziel dieses Beitrags ist die Explizierung von Zielen und Kennzahlen für die optimale Integration von Applikationen und für die Steuerung der Applikationsarchitektur. Grundlage dafür sind zunächst generelle Überlegungen zum „optimalen Integrationsgrad“ (Abschnitt 2). In Abschnitt 3 werden allgemeine Eigenschaften von Kennzahlen und Kennzahlensystemen untersucht, die auch für die Planung und Steuerung der Applikationsarchitektur relevant sind. Eine Untersuchung der Literatur im Hinblick auf Erfolgsfaktoren für die Applikationsintegration erfolgt in Abschnitt 4. Zu den jeweiligen Zielen werden Kennzahlen vorgeschlagen, anhand derer die Zielerreichung gemessen werden kann. Schließlich werden in Abschnitt 5 komplementäre und konkurrierende Wechselwirkungen zwischen den einzelnen Zielen qualitativ aufgezeigt. Das Ergebnis dieses Beitrags stellt Hypothesen für die Beziehungen innerhalb des Zielsystems sowie Vorschläge für geeignete Kennzahlen dar. Diese Ergebnisse müssen in weiterführenden Arbeiten qualitativ und/oder quantitativ validiert werden.

2 Applikationsintegration

Aus technischer Sicht kann eine Applikation als eine aggregierte Zusammenfassung bestimmter Softwareartefakte (z.B. Module, Komponenten, Datenstrukturen) aufgefasst werden, die in einem engen Zusammenhang stehen (z.B. Aufruf, Zugriff). In diesem Beitrag wird jedoch die fachliche Sicht eingenommen: In dieser repräsentiert eine Applikation eine aggregierte Zusammenfassung bestimmter Funktionalitäten, die über gemeinsam unterstützte Geschäftsprozesse, gemeinsam genutzte Informationen, gemeinsame Wiederverwendung und/oder gemeinsame Verantwortlichkeiten in einem engen fachlichen Zusammenhang stehen.

Bei der Applikationsgestaltung führen enge Zusammenhänge im Normalfall dazu, dass die betroffenen Funktionalitäten in einer gemeinsamen Applikation zusammengefasst werden. Weniger enge Zusammenhänge werden in Form von Schnittstellen zwischen Applikationen abgebildet. Je nach Abgrenzung „enger“ und „schwacher“ Zusammenhänge entsteht somit eine unterschiedliche Anzahl von Applikationen und Schnittstellen.

Je weniger Applikationen gebildet werden, desto weniger Schnittstellen sind abzubilden und desto geringer fallen Aufwendungen für die Entwicklung und den Betrieb dieser Schnittstellen aus. Dieser Zusammenhang wird durch die Kurve „Schnittstellenkosten“ in Abbildung 1 illustriert. Aufgrund des Netzeffekts darf erwartet werden, dass diese Kurve einen mit zunehmender Anzahl von Applikationen exponentiell steigenden Verlauf hat. Je weniger Applikationen gebildet werden, desto höher werden jedoch die Aufwendungen zur Entwicklung und zum Betrieb dieser – entsprechend komplexeren – Applikationen sein [Wint05]. Dieser Zusammenhang wird durch die Kurve Applikationskosten („Anwendungssystemkosten“) in Abbildung 1 illustriert. Auch hier darf ein mit abnehmender Anzahl von Applikationen exponentiell steigender Verlauf unterstellt werden. In Abbildung 1 sind beide Kurven in Anlehnung an die Darstellung bei [Fres95] (zitiert nach [Rose99, S. 14]) dargestellt.

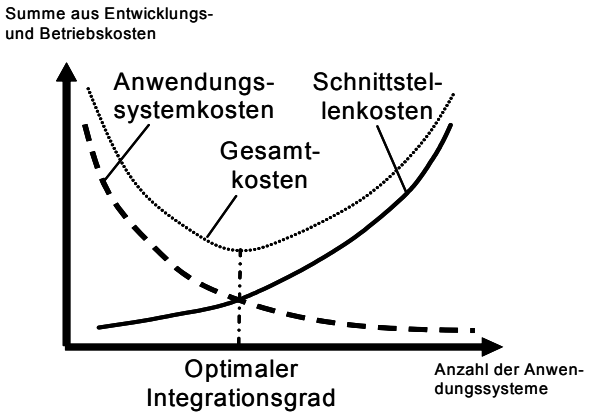

Abbildung 1: Optimaler Integrationsgrad (vgl. [Wint05])

Leider kann der „optimale Integrationsgrad“ in der Realität aufgrund unstetiger Kostenverläufe, ungenauer Betriebskostenschätzungen, technischer Integrationsprobleme etc. nicht analytisch bestimmt und vor allem nicht durch entsprechende Applikationsgestaltung umgesetzt werden. Deshalb muss versucht werden, durch geeignete Steuerungsmechanismen solche Applikationsentwicklungsprojekte durchzuführen, die die Ist-Situation in Richtung des „optimalen Integrationsgrads“ verbessern, d.h. die die Summe aus Entwicklungs- und Betriebskosten reduzieren. Ein entsprechend konsequentes Projektportfoliomanagement kann aber nur erfolgen, wenn die Beiträge einzelner Applikationsentwicklungsprojekte zur Erfüllung des Integrationsziels klar messbar sind und wenn die Beziehungen zwischen verschiedenen Integrationszielen klar sind. Das in diesem Beitrag vorgeschlagene Zielsystem dient genau diesem Zweck. Dies erfordert eine weitere Detaillierung der Applikationskosten und Schnittstellenkosten, da diese Größen nicht unmittelbar bestimmt werden können. Bevor Kennzahlen zur qualitativen Bestimmung

der genannten Größen entwickelt werden, werden allgemeine Eigenschaften von Kennzahlen vorgestellt.

3 Kennzahlen und Kennzahlensysteme

[Reic97] definiert Kennzahlen als Zahlen, die quantitativ erfassbare Sachverhalte in konzentrierter Form erfassen. Ein Kennzahlensystem ist eine geordnete Gesamtheit von einzelnen Kennzahlen, die in einer Beziehung zueinander stehen und so als Gesamtheit über einen Sachverhalt vollständig informieren. Kennzahlen dienen zur Verdichtung großer Datenmengen zu wenigen, aussagekräftigen Kenngrößen [Jäge01, S. 127]. Sie sind Hilfsmittel für die Planung, Steuerung und Kontrolle. Zentrale Eigenschaften einer Kennzahl sind nach [Jäge01, S. 127]:

- Informationscharakter: Kennzahlen sollen eine Beurteilung wichtiger Sachverhalte ermöglichen.
- Quantifizierbarkeit: Die Sachverhalte müssen auf einer metrischen Skala gemessen werden können.

Der Einsatz von Kennzahlen verlangt, dass Messdaten auch wirklich vorhanden sind und systematisch erfasst werden können. Nur so können Sachverhalte genau quantifiziert werden. Weiterhin werden in der Literatur folgende Anforderungen an eine sinnvolle Einsetzbarkeit von Kennzahlen im Informationsmanagement gestellt [Jäge01]:

- Zweckeignung: Der Informationsgehalt der Kennzahl soll möglichst mit dem ursprünglichen Informationsbedarf übereinstimmen. Ein hoher Deckungsgrad zwischen Informationsbedarf und Informationsangebot stellt sicher, dass nicht unbrauchbare Kennzahlen entwickelt werden.
- Genauigkeit: Die Anforderung an die Genauigkeit einer Kennzahl wird durch ihre Reliabilität (Zuverlässigkeit) und Validität (Gültigkeit) bestimmt.
- Aktualität: Der Zeitraum zwischen der Messung der Kennzahl und ihrer Auswertung sollte möglichst gering sein.
- Kosten-Nutzen-Relation: Die Erhebung einer Kennzahl sollte nicht höhere Kosten verursachen, als ihr Erkenntniswert ist [Hauf89].
- Einfachheit und Nachvollziehbarkeit: Der Verwender der Kennzahl muss das Messergebnis einfach interpretieren können und das Messergebnis muss zurückverfolgbar sein.

Im folgenden Abschnitt werden Ziele bei der Gestaltung der Applikationsintegration erarbeitet. Anschließend werden zu den einzelnen Zielen Kennzahlen diskutiert, die eine Operationalisierung der Zielerreichung ermöglichen können. Bei der Entwicklung der Kennzahlen werden die aufgezeigten Anforderungen an Kenn-

zahlen berücksichtigt. Neben dem im Folgenden vorgestellten Ansatz existieren zahlreiche Kennzahlensysteme im Informationssystemmanagement, jedoch findet in ihnen keine Fokussierung auf die Applikationsintegration statt. Ein ausführlicher Überblick von Führungsinstrumenten für Informationssysteme ist in [Jäge01] zu finden.

4 Ziele bei der Gestaltung der Applikationsintegration

Bei der Applikationsintegration ist der direkte Einfluss auf Finanzergebnisse kaum zu belegen. Die in Abschnitt 2 aggregiert betrachteten Größen „(gesamte) Applikationskosten“ und „(gesamte) Schnittstellenkosten“ eignen sich nicht, um Detailentscheidungen in einzelnen Applikationsentwicklungsprojekten treffen zu können. Zudem ist unbestritten, dass Architekturmanagement nicht nur zur Kostenoptimierung beiträgt, sondern auch die Wartbarkeit, die Entwicklungsgeschwindigkeit und die Flexibilität erhöhen muss [Haf+04].

In der Literatur finden sich zahlreiche Ansätze zur Applikationsintegration, die auch häufig unter dem Stichwort Enterprise Application Integration (EAI) diskutiert werden. Diese wurden neben ausgewählten Ansätzen aus der Unternehmenspraxis hinsichtlich der Zielkriterien bei der Applikationsintegration untersucht. Tabelle 1 fasst zusammen, welche Ziele der Applikationsintegration in den Ansätzen aus der Literatur (oberer Teil) und den Praxisansätzen (unterer Teil) verfolgt werden. In der Literatur sind vor allem fünf Ziele zu finden, die teilweise explizit, aber teilweise auch nur implizit genannt wurden:

- Minimale Projektaufwände (Zeit und Kosten) für die Integration von Applikationen in die vorhandene Applikationslandschaft
- Optimale Wiederverwendung von Komponenten und minimale funktionale Redundanz
- Komplexitätsreduktion innerhalb der Applikationslandschaft
- Optimaler Kopplungsgrad von Applikationen („Kopplungsgrad“ wird verwendet, um die Verwechslung dieser Zielgröße mit der Makrobetrachtung in Abschnitt 2 zu verhindern)
- Minimale Infrastrukturkosten und –komplexität

Ansatz/Kriterium	Minimale Projektaufwände	Optimale Wiederverwendung	Komplexitätsreduktion	Optimaler Kopplungsgrad	Minimale Infrastruktur-kosten/-omplexität
Ansätze aus der Literatur					
[Lint00, S. 9, S. 25, S. 61]		■	■	■	
[Zaha00, S. xliv, S. 5, S. 79]	■	■	■		■
[Kaib02, S. 22ff.]	■	■	■	■	■
[Ruh+01, S. 6ff., S. 12, S. 20, S. 156]	■	■	■	■	■
[Cumm02, S. 423ff.]	■	■	■	■	■
[FrHe04]			■		■
[ThIr01, S. 327f.]	■	■		■	■
Ansätze aus der Praxis					
[Lisk03, S. 10]	■		■		
[Moll03, S. 12, S. 24]	■		■		■
[KuSc03, S. 6, S. 12]	■		■	■	■
[Bath03, S. 5]]	■	■	■	■	
[Endr03, S. 13, S. 18]			■		■
[Grög03, S. 3]	■	■	■		■
[Knec03, S. 9/10]		■		■	■
[Hofe03, S. 26]	■	■	■		
[Frie03, S. 23]		■	■		■
[Aust03, S. 4, S. 19]	■		■	■	■

Tabelle 1: Ziele bei der Applikationsintegration aus Literatur und Praxis

Zur näheren Analyse dieser Ziele wird eine zentrale Größe in den Mittelpunkt gestellt, von der angenommen werden darf, dass sie von den einzelnen Teilfaktoren beeinflusst wird. In Anlehnung an den in letzter Zeit häufig verwendeten Begriff der Agilität eines Unternehmens (d.h. der Fähigkeit zur raschen Anpassung an wechselnde Anforderungen) wird der Begriff der Agilität eines Informationssystems analog definiert:

Die Agilität eines Informationssystems stellt die Fähigkeit dar, auf alle Arten von Veränderungen das Informationssystem effektiv und effizient anpassen zu können [AmMo04]. Letztendlich muss es das Ziel der Applikationsarchitektur sein, diese Agilität zu steigern und damit die Widerstände der Applikationslandschaft gegen Änderungen oder Erweiterungen zu reduzieren. Im Folgenden sollen Effizienzkriterien identifiziert werden, die bei Integrationsfragestellungen unmittelbaren Einfluss auf die Agilität eines Informationssystems haben. Es wäre durchaus denkbar, andere Größen in den Mittelpunkt zu stellen – beispielsweise kostenorientierte Größen – jedoch wird in diesem Beitrag die Agilität in den Mittelpunkt gestellt, da sie nicht nur rein kostentechnische Aspekte betrachtet, die schwer direkt zugeordnet werden können.

Die Literatur liefert häufig nur verallgemeinerte Aussagen zum Nutzen einer Integrationslösung (z.B. Kostenreduktion durch Senkung der Anzahl der zu wartenden Schnittstellen durch Einführung einer Bus-Architektur). Die quantitative Messung bzw. die Vorgabe von Messgrößen fehlt jedoch meist. Zunächst werden die einzelnen Ziele detaillierter beschrieben. Anschließend wird auf Wechselwirkungen einzelner Ziele und auf Zielkonflikte eingegangen. Das Gesamtzielssystem wird in Abbildung 2 dargestellt, worin auch die Abhängigkeiten einzelner Teilziele erkennbar sind.

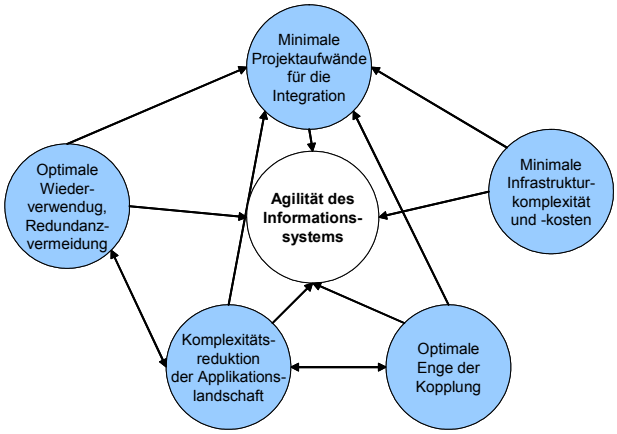

Abbildung 2: Ziele und Wechselwirkungen bei der Applikationsintegration (in Anlehnung an [ScHa04])

4.1 Agilität des Informationssystems

Die Agilität des Informationssystems drückt die „Leichtigkeit" aus, mit der sich Änderungen oder Erweiterungen durchführen lassen. Diese Änderungen und Erweiterungen können sowohl technischer (z.B. Auswechseln einer Applikation aufgrund auslaufender Wartungsverträge) wie auch organisatorischer Natur sein (z.B. Outsourcing bestimmter Applikationen oder Prozesse). Viele Faktoren können

diese Agilität beeinflussen – die Informationssystem-Architektur ist nur einer von ihnen. Es wird jedoch davon ausgegangen, dass die Informationssystem-Architektur, speziell die Applikationsarchitektur, mit steigender Größe des Informationssystems einen zunehmend signifikanten Einfluss auf die Agilität hat. Dies ist insbesondere auf den Wandel zurückzuführen, der sich in den letzten Jahrzehnten vollzogen hat: Bis Ende der 80er Jahre war die Entwicklung geprägt durch die Eigenentwicklung monolithischer Applikationen in nahezu allen betrieblichen Bereichen – unabhängig davon, inwieweit die durch die jeweiligen Applikationen unterstützten Aufgaben Kernkompetenzen darstellten. Es ging vorwiegend um die großflächige Umstellung auf „EDV". Der ersten Phase folgte der Trend zum Einsatz von Standardsoftware, der bis Ende der 90er Jahre anhielt. Hoch integrierte Standardsoftware (z. B. SAP R/3) wurde eingeführt, die ein breites Spektrum betrieblicher Aufgaben unterstützt. Die rasante Entwicklung des E-Business der vergangenen Jahre ließ – ähnlich wie in der ersten Phase – eine Vielzahl neuer Applikationen (v.a. Internetapplikationen) entstehen. Oft stand die schnelle Produktivsetzung im Vordergrund, wodurch eine kontrollierte und geführte Weiterentwicklung der Applikationslandschaft vernachlässigt wurde. Häufig wurde die benötigte Funktionalität neu programmiert ohne vorhandene Funktionalität über Schnittstellen zu nutzen. Nach dieser dritten Phase gilt es, Applikationen zu konsolidieren, entstandene Redundanzen zu beseitigen und sie in eine einheitliche Architektur zu überführen. Zwar ist eine Konsolidierung von Applikationen zu erkennen, jedoch müssen neu hinzu kommende Applikationen möglichst effektiv und effizient in die vorhandene Applikationslandschaft integriert werden. Eine Untersuchung der Gartner Group hat ergeben, dass ca. 30% der Zeit bei Softwareentwicklungsprojekten für die Integration (Schnittstellendesign und Schnittstellenimplementierung) verwendet werden muss. Eine weitere Untersuchung der Gartner Group hat ergeben, dass durchschnittlich 40% der IT-Budgets für die Erstellung und Wartung von Schnittstellen ausgegeben werden [Kral03, S. 8].

Direkt messen lässt sich der Einfluss der Applikationsarchitektur auf die Agilität des Gesamtsystems heute schwer – Bedingung dafür wären genauere Analysen der Zusammenhänge zwischen der Komplexität der Inter-Applikations-Strukturen und der Agilität.

Die Agilität selbst ist aber prinzipiell messbar, wenn auch zeitverzögert und nur mit einigem Aufwand. Die Grundidee ist, die Gesamtaufwände für Änderungen oder Erweiterungen am Informationssystem in einem bestimmten Zeitraum zu messen und sie anhand der Größe der durchgeführten Änderungen zu normieren. Grund für die Betrachtung eines längeren Zeitraums und nicht die Betrachtung von Einzelprojekten liegt in der Natur des Architekturmanagements, da sich die Schaffung von Architekturkonformität zumindest nicht kurzfristig rechnet („You Can't 'Cost-Justify' Architecture" [Zach01]).

Die Erfassung von Größen stellt allerdings einige Anforderungen an das IT-Controlling eines Unternehmens, denn zum einen müssen die Integrationsaufwän-

de (Kosten) und die Integrationsleistung aus Gesamtprojekten extrahiert und normiert werden.

Im Folgenden werden auf Grundlage der Literatur- und Praxisanalyse am Beginn dieses Abschnitts fünf Erfolgsfaktoren und mögliche Kennzahlen diskutiert, die offensichtlich wesentlichen Einfluss auf die Agilität des Gesamtsystems haben.

4.2 Komplexitätsreduktion durch Desintegration der Applikationslandschaft

Historisch gewachsene Applikationslandschaften mit vielen hundert Applikationen (vgl. [Lint00, S. 6ff.]) lassen sich schwer als Ganzes steuern. Dazu sind die verschiedenen Abhängigkeiten und Querbeziehungen zwischen den Applikationen zu komplex. Es ist daher notwendig, die Komplexität der Applikationslandschaft beherrschbar zu machen, indem gezielt „Desintegration“ betrieben wird. Unter Komplexität der Applikationslandschaft wird in diesem Zusammenhang die Eigenschaft verstanden, die die Beschreibungen ihres Gesamtverhaltens in einer beliebigen Sprache erschwert, selbst wenn man vollständige Informationen über einzelne Applikationen und ihrer Wechselwirkungen besitzt. Die Applikationslandschaft wird deshalb in handhabbare Einheiten zerlegt, zwischen denen eine nur lose Kopplung existiert.

Eine Möglichkeit besteht darin, Applikationsdomänen zu bilden [ScHa04]. Es handelt sich dabei um Gruppierungen von Applikationen und Informationsobjekten, die einem gemeinsamen Fachbereich zugerechnet werden können. Die Anzahl der zu bildenden Domänen sollte allerdings klein bleiben, um eben den Vorteil der Komplexitätsreduktion zu wahren. In einem konkreten Anwendungsfall bei einem Schweizerischen Finanzdienstleistungsunternehmen wurden ca. 20 solcher Applikationsdomänen definiert [ScHa04]. Ziel der Applikationsintegration ist es, innerhalb dieser Domänen eng zu koppeln und außerhalb der Domänen lose. Der Effekt ist eine Reduktion von Abhängigkeiten zwischen Applikationen in unterschiedlichen Domänen und damit die Möglichkeit, einzelne Domänen autonom weiterzuentwickeln, ohne dass andere Domänen direkt betroffen sind.

Auch wenn die organisatorischen Verantwortlichkeiten für Applikationsdomänen bereits definiert sind, werden in der Regel noch eng gekoppelte Abhängigkeiten zwischen den Bereichen/Domänen bestehen. Dies ist insbesondere der Fall, wenn eine existierende Applikationslandschaft „desintegriert“ werden muss. Die Reduktion direkter Abhängigkeiten wird viele Jahre in Anspruch nehmen [ScHa04].

Die wichtigste Steuerungsgröße ist deshalb der Grad der Desintegration zwischen den Applikationsdomänen. Um diese Größe zu erfassen, muss die Anzahl lose gekoppelter kontrollierter Verbindungen zwischen Domänen in Relation zur Gesamtzahl von Verbindungen zwischen den Domänen gesetzt werden. Der entstehende Quotient gibt Auskunft über die Entkopplung der Landschaft. Verfeinerte

Werte für bestimmte Bereiche oder Applikationsgruppen können dann Aufschluss über den dringendsten Handlungsbedarf geben.

Voraussetzung für die Erhebung geeigneter Kennzahlen sind jedoch technische Hilfsmittel, mit Hilfe derer sich die Beziehungen zwischen Applikationen analysieren lassen. Entsprechende Werkzeuge existieren am Markt. Sie analysieren entweder den Source-Code der Applikationen oder führen Messungen zur Laufzeit durch, um die gewünschten Informationen zu beschaffen. Source-Code-Analysen haben den Vorteil, dass sie die Performance der produktiven Applikationen nicht reduzieren. Sie setzen jedoch recht homogene Entwicklungsumgebungen voraus.

4.3 Optimale Enge der Kopplung

Die „optimale“ Kopplung im architektonischen Sinne realisiert für jede Applikationsbeziehung einen Grad der Integration, der weder zu eng noch zu lose ist. Dies kann für unterschiedliche Applikationen sehr unterschiedliche Kopplungen zur Folge haben. Generell gilt: Je „näher“ zwei Applikationen zueinander sind, desto enger wird man koppeln, um Reibungsverluste zu vermeiden. Auf der anderen Seite wird man umso loser koppeln, je „weiter“ zwei Applikationen voneinander entfernt sind. Die optimale Kopplung hat direkten Einfluss auf die Agilität, da bei zu enger Kopplung die Aufwände für Änderungen steigen. Bei zu loser Kopplung steigen Kommunikationskosten (sowohl menschliche, z.B. zwischen zwei Abteilungen, als auch maschinelle, z.B. zusätzliche Middleware muss eingesetzt werden oder zusätzliche Transformationen sind erforderlich) und es entsteht dadurch vor allem ein Overhead zur Laufzeit.

Das Problem ist hier, dass der optimale Fall für jede Applikationsbeziehung unterschiedlich ist. Objektive Kriterien oder Verfahren zur Ermittlung des richtigen Kopplungsgrads sind nicht bekannt. Es ist daher auch sehr schwer, Kennzahlen und Messgrößen anzugeben. Indikatoren wären auf der einen Seite die tatsächlichen Kosten für Anpassungen an Schnittstellen oder ganzen Applikationen, aufgrund von Änderungen in gekoppelten Applikationen (sehr hohe Kosten deuten auf eine zu enge Kopplung hin). Auf der anderen Seite müsste der Entwicklungs- und Laufzeitoverhead für lose gekoppelte Systeme gemessen werden (z.B. die für Middleware aufgewendeten MIPS). Diese Zahlen sind jedoch üblicherweise in den Projektentwicklungs- bzw. Unterhaltskosten versteckt und daher nicht direkt messbar.

Selbst wenn sie zu beschaffen wären, ergäbe sich darüber hinaus das Problem der richtigen Bewertung. Benchmark-Studien scheinen hier der einzige Weg zu sein, um zu aussagekräftigen Steuergrößen zu kommen. Diese fehlen jedoch, vor allem, da das Thema der Steuergrößen für die Applikationsarchitektur noch nicht genügend weit etabliert ist.

Als Alternative sind Stichproben an einzelnen Applikationen denkbar. Analysen der Änderungshäufigkeit dieser Applikationen aufgrund von Änderungen im Umfeld sowie Messungen der für die lose Integration anfallenden Laufzeitkosten (beides in Relation zur Komplexität der jeweiligen Applikation) können Anhaltspunkte über einen guten oder schlechten Kopplungsgrad liefern. Das Problem der richtigen Bewertung der Zahlen besteht aber auch hier.

4.4 Optimale Wiederverwendung, Redundanzvermeidung

Das Ziel der optimalen Wiederverwendung fordert, dass jede Funktionalität nur einmal implementiert und dass jede Information nur einmal gespeichert werden sollte. Im Idealfall lassen sich so die Entwicklungs- und Unterhaltskosten minimieren. Zudem fördert Wiederverwendung die Konsistenz, die Qualität und die Flexibilität von Applikationen [Cumm02, S. 430]. Um dieses Ziel zu erreichen, muss Middleware zur Verfügung stehen, um die zentral implementierte Funktionalität von unterschiedlichen technischen Plattformen aus zu erreichen, und geeignete Design-Grundlagen müssen die Wiederverwendbarkeit der Funktionen sicherstellen.

Bereits bei der Entwicklung potenziell wiederverwendbarer Komponenten (z.B. Services) muss darauf geachtet werden, dass sie bestmöglich wiederverwendbar sind (Design-for-Reuse). Eine wichtige Fragestellung ist dabei die gewählte Granularität: Werden sehr große monolithische Komponenten mit umfassender Funktionalität entwickelt, können diese oder Teile davon aufgrund ihrer umfangreichen Funktionalität sehr häufig wiederverwendet werden, jedoch entstehen auch zahlreiche Abhängigkeiten. Beispielsweise sind Releasezyklen kürzer, da häufiger Änderungen in umfassenden Komponenten vorgenommen werden müssen als in modularen. Werden auf der anderen Seite die Komponenten zu feingranular entwickelt, reduziert sich der Wiederverwendungsgrad und es entsteht Zusatzaufwand bei der Wartung und bei der Ausführung zur Laufzeit.

Ein weiteres Kriterium ist der Spezialisierungsgrad von Komponenten. Werden sehr spezialisierte Komponenten entwickelt, ist das Potenzial, dass diese wiederverwendet werden, eher gering, da der Anwenderkreis dieser sehr spezialisierten Komponenten beschränkt sein wird. Werden auf der anderen Seite Komponenten zu allgemein entwickelt, kann zwar die Wiederverwendungsquote höher sein, jedoch entsteht Zusatzaufwand für die Implementierung zusätzlicher Logik.

Zur Messung der Wiederverwendung ist die wichtigste Kennzahl die durchschnittliche Wiederverwendung pro zentraler Funktion. Zur Ermittlung dieser Zahl sind Repositories notwendig, die über Anzahl und Nutzung der verschiedenen Komponenten und ihrer Schnittstellen Auskunft geben. Dazu könnten ähnliche Tools, wie die im vorigen Kapitel erwähnten Source-Code-Analysierer, eingesetzt werden.

Weitere wichtige Kennzahlen, die Anhaltspunkte bzgl. der Güte der Applikationsarchitektur geben können, sind die Gesamtzahl publizierter Schnittstellen von Komponenten sowie deren Wachstum. Eine sehr große Anzahl und ein starkes Wachstum sollten als mögliche Anzeichen für zu große Redundanzen Berücksichtigung finden. Allerdings gibt es hier bislang keine Erkenntnisse über die „ideale" Zahl wiederverwendbarer Funktionen. Zudem lässt ein rasantes Wachstum von wieder verwendbaren Komponenten nicht notwendigerweise auf die Entstehung von Redundanzen schließen. Bei der Einführung neuer Technologien oder neuer Architekturkonzepte (z.B. Umstellung auf eine serviceorientierte Architektur) kann eine rasant steigende Anzahl von Services auch auf hohe Nutzerakzeptanz zurückzuführen sein. Bei der Betrachtung derartiger Kennzahlen sind also auch solche Nebenbedingungen zu berücksichtigen.

4.5 Minimale Projektaufwände für die Integration

Die Bewertung der Integrationskosten von Projekten ist problematisch, weil die Integrationskosten nicht absolut gewertet werden können, sondern in Relation zum jeweiligen zugrunde liegenden Integrationsproblem gesetzt werden müssen. Eine isolierte Applikation mit nur wenigen Beziehungen zu anderen Applikationen hat wesentlich geringere Integrationskosten als eine Applikation, die viele Daten von sehr unterschiedlichen Systemen beziehen muss. Zudem muss berücksichtigt werden, dass einzelne frühe Projekte Komponenten implementieren, die später von anderen Projekten wiederverwendet werden können. Die Integrationskosten werden im ersten Projekt naturgemäß am höchsten sein. Deshalb dürfen nicht Einzelprojekte betrachtet, sondern es muss das gesamte Projektportfolio über einen längeren Zeitraum hinweg untersucht werden.

Rückschlüsse auf die Qualität der Integrationsaspekte der Applikationsarchitektur lassen sich damit aus den Integrationskosten nur ziehen, wenn diese gewichtet werden. Sie sind abhängig von zahlreichen schwer determinierbaren Größen wie Anzahl der Schnittstellen, beteiligte Organisationseinheiten, Zustand der Dokumentation, unterschiedliche Semantik, Technik, usw. Dies bedeutet, dass eine Messung der Integrationskosten die Komplexität des Integrationsproblems der jeweiligen Applikation berücksichtigen müsste. Geeignete Bewertungsverfahren für die Integrationskomplexität könnten auch helfen, unterschiedliche Integrationsmethoden empirisch in Relation zu setzen.

Ohne ein solches Maß wäre es nur möglich, eine vergleichbare Alternative bei der „Implementierung der Integration" zu betrachten. Eine Alternativimplementierung ist aber in der Regel aus Kostengründen nicht möglich und wird außer Betracht gelassen.

Damit ist der einzige gangbare Weg ein Vergleich der gewichteten Integrationskosten über mehrere Zeiträume hinweg. Dazu können die gesamten Integrationskosten eines Jahres (Summe über alle Projekte) betrachtet werden und durch die

Gesamt-Integrationskomplexität dividiert werden. Dividiert man nun den entsprechenden Bruch aus dem zweiten Vergleichsjahr, sollte das Ergebnis größer 1 sein. Dies würde ausdrücken, dass die Kosteneffizienz bei vergleichbarer Integrationskomplexität verbessert wurde, also mehr Integrationskomplexität mit weniger Aufwand implementiert worden ist.

Ungelöst bleibt zunächst das Problem des anzustrebenden Zielzustandes. Dieser lässt sich ohne industrieweite Benchmarks nur schwer definieren.

4.6 Minimierung der Infrastrukturkosten und -komplexität

Die Anzahl der eingesetzten Integrationstechnologien hat direkten Einfluss auf die fixen IT-Kosten (da z. B. für jedes Tool Personal bereitstehen muss) und in gewissem Maße auch auf die Implementierungskosten: Je weniger Tools existieren, desto besser können diese unterstützt werden. Bei Einsatz einer großen Anzahl steigt die Komplexität (im Sinne der Definition aus Abschnitt 4.2) beispielsweise dadurch, dass die Unsicherheit bei den Entwicklern, welches Werkzeug in welcher Situation einzusetzen ist steigt, wodurch wiederum in jedem Projekt Aufwände für Analysen notwendig werden.

Auf der anderen Seite ist aber eine gewisse Grundmenge an Technologien notwendig, um „Workarounds“, d.h. die Simulation einer Technologie durch eine andere, zu verhindern. Als Beispiel sei hier der „Nachbau“ einer Servicearchitektur durch messageorientierte Middleware genannt. Ein solches Vorgehen bringt auf Dauer nur Nachteile, da spezifische Eigenschaften (im Beispielfall z.B. die Zuordnung von Aufrufen zu Ergebnissen und die Fehlerbehandlung, falls Antworten ausbleiben), die in der benutzten Middleware nicht vorhanden sind, aufwändig selbst implementiert werden müssen. Ziel ist es, mit möglichst wenigen Standard-Technologien möglichst viele Anforderungen zu erfüllen.

Mögliche Kennzahlen zur Steuerung dieses Ziels sind neben den Infrastrukturkosten beispielsweise die Anzahl eingesetzter Technologien, der Grad des Einsatzes einheitlicher Produkte oder der Erfüllungsgrad der technologischen Anforderungen in Projekten.

5 Wechselwirkungen einzelner Ziele

Es ist offensichtlich, dass die verschiedenen in Abschnitt 4 beschriebenen Ziele untereinander Wechselwirkungen haben. Teilweise sind Ziele offensichtlich komplementär, andere offensichtlich konkurrierend. Im Folgenden werden die zehn möglichen Wechselwirkungen zwischen den fünf Zielen qualitativ untersucht. Die daraus resultierenden Hypothesen bilden die Grundlage für eine quantitative Analyse in weiterführenden Arbeiten.

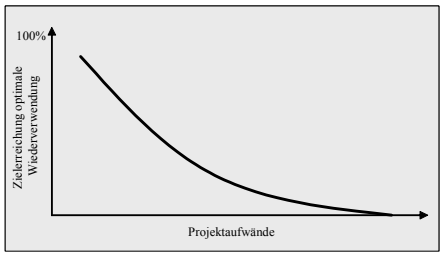

Abbildung 3

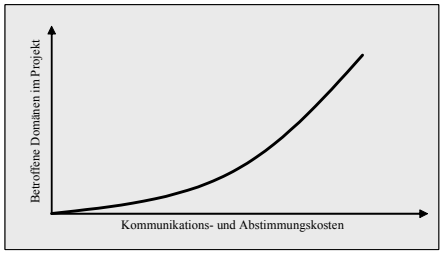

Abbildung 4

Wechselwirkungen zwischen Minimierung der Projektaufwände und optimaler Wiederverwendung (vgl. Abbildung 3)

Zum einen können Projektaufwände minimiert werden, wenn möglichst viele, bereits vorhandene Komponenten wiederverwendet werden können, zum anderen müssen Aufwände in Projekten getätigt werden, die ohne das Ziel der optimalen Wiederverwendung hinfällig wären. Denn die Entwicklung wiederverwendbarer Komponenten ist in der Regel aufwändiger als die Entwicklung von Einwegkomponenten [Boe+00], [Ruh+01, S. 134f.]. Dies ist vor allem auf den erhöhten Aufwand beim Design und bei der Qualitätssicherung zurückzuführen. Bei der Entwicklung wiederverwendbarer Komponenten müssen Schnittstellen so entworfen werden, dass sie auch von zukünftigen Projekten wiederverwendet werden können.

Da das Ziel der Minimierung von Projektaufwänden nicht die Minimierung der Aufwände eines einzelnen Projekts, sondern die Minimierung der Aufwände des gesamten Projektportfolios über einen längeren Zeitraum hinweg zum Ziel hat, beeinflusst das Ziel der Wiederverwendung das Ziel der Minimierung der Projektaufwände positiv, da die Einsparungen über einen längeren Zeitraum höher sein sollten als die initialen Entwicklungskosten. Dies setzt allerdings voraus, dass die initial entwickelten Komponenten auch tatsächlich wiederverwendet werden. Daneben muss berücksichtigt werden, dass auch die Wiederverwendung Kosten erzeugt, die ohne nicht anfallen würden. Beispielsweise muss Know-How aufgebaut werden, um die Semantik einer vorhandenen Komponente zu verstehen.

Wechselwirkungen zwischen Minimierung der Projektaufwände und Komplexitätsreduktion (vgl. Abbildung 4)

Um dem Ziel der Komplexitätsreduktion nachzukommen, muss durch gezielte Desintegrationsmaßnahmen eigenständige Domänen entwickelt werden, die autonom von anderen sind. Werden Projekte innerhalb einer Domäne abgewickelt, ist der Einfluss auf die Projektaufwände gering. Wird aber ein Projekt durchgeführt, das mehrere Domänen betrifft, steigt unter Umständen der Aufwand, da organisatorische Hürden überwunden werden müssen. Der Eingriff in eine autonome Domäne erfordert großen Abstimmungsbedarf zwischen verschiedenen Organisationseinheiten. Der Abstimmungsbedarf wäre zwar ohne die Bildung von Domänen ebenso gegeben, jedoch wären die Kommunikationswege kürzer und unkompli-

zierter. Es wird angenommen, dass der Kommunikationsaufwand mit zunehmender Anzahl beteiligter Domänen exponentiell steigt, da nicht nur die Projektinitiierende Domäne mit den beteiligten kommunizieren muss, sondern auch die anderen beteiligten untereinander.

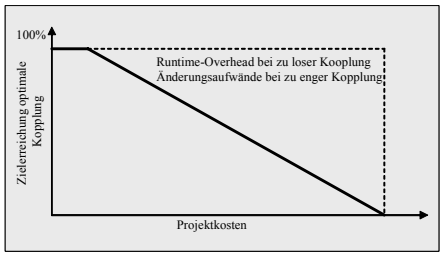

Abbildung 5

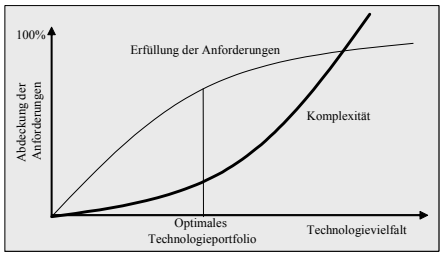

Abbildung 6

Wechselwirkungen zwischen Minimierung der Projektaufwände und optimalem Kopplungsgrad (vgl. Abbildung 5)

Das Ziel der optimalen Kopplung will zum einen die Abhängigkeiten zwischen Applikationen minimieren und gleichzeitig den Entwicklungs- und Runtime-Overhead vermeiden. Wird ein Projekt durchgeführt, das eng gekoppelte Applikationen betrifft, sollte bei Erreichung der optimalen Kopplung auch die Anzahl betroffener Applikationen – und damit die Änderungsaufwände – minimal sein [Lint00, S. 25]. Sind in einem Projekt lose gekoppelte Applikationen beteiligt, so sollte durch Erreichung des Ziels der optimalen Kopplung auch der Entwicklungs- und Runtime-Overhead minimiert werden. Die beiden Ziele sind somit komplementär zueinander.

Wechselwirkungen zwischen Minimierung der Projektaufwände und minimaler Infrastrukturkomplexität und -kosten (vgl. Abbildung 6)

Um dem Ziel der minimalen Infrastrukturkomplexität nahe zu kommen, muss eine Beschränkung auf wenige Infrastrukturkomponenten erfolgen. Dies hat eine Einschränkung nutzbarer Technologien zur Folge (z.B. bei einer serviceorientierten Architektur Beschränkung auf Corba-Services ohne Option auf Web Services). Diese wiederum impliziert, dass zwar die Infrastrukturkosten (beispielsweise in Form von Lizenzgebühren, Supportkosten, Kosten für den Aufbau von Know-How für neue Technologien, usw.) und -komplexität gering bleiben, Projekte jedoch auf die Verwendung vorgegebener (im Unternehmen eingesetzter) Technologien eingeschränkt sind. Deshalb sind in Projekten evtl. Zusatzaufwände notwendig, die beim Einsatz einer anderen Technologie, die die Anforderungen des Projekts besser erfüllen würde, nicht nötig wären. Die beiden Ziele konkurrieren also miteinander.

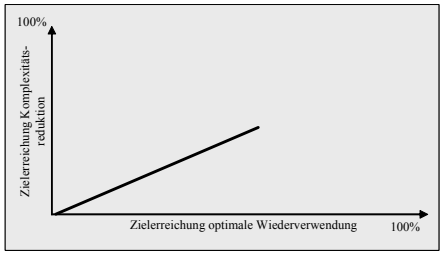

Abbildung 7

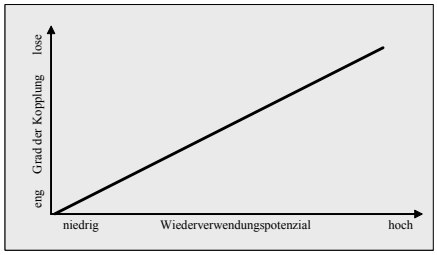

Abbildung 8

Wechselwirkungen zwischen optimaler Wiederverwendung und Komplexitätsreduktion (vgl. Abbildung 7)

Ähnlich wie bei Projekten ist auch hier das Hauptproblem im Kommunikations- und Abstimmungsbedarf zu sehen, sofern Komponenten entwickelt werden, die von mehreren Domänen verwendet werden. Hier müssen Fragestellungen bezüglich der Granularität, der Generalisierung und der Spezialisierung der zu entwickelnden wieder verwendbaren Komponente geklärt werden. Allgemein fördert eine optimale Wiederverwendung die Komplexitätsreduktion [Ruh+01, S. 134ff.], jedoch impliziert eine optimale Wiederverwendung keine optimale Komplexitätsreduktion oder umgekehrt.

Wechselwirkungen zwischen optimaler Wiederverwendung und optimalem Kopplungsgrad (vgl. Abbildung 8)

Um einen möglichst hohen Wiederverwendungsgrad zu erreichen, müssen Applikationen in der Regel lose gekoppelt werden. Ein hoher Wiederverwendungsgrad besagt, dass dieselben Komponenten von vielen unterschiedlichen Applikationen genutzt werden. Wären diese Applikationen eng miteinander gekoppelt, wäre der Änderungsaufwand bei Änderung einer Komponente zu groß. Auf der anderen Seite sind eng gekoppelte Komponenten nicht auf Wiederverwendbarkeit hin ausgerichtet. Wiederverwendung setzt Kopplung von Applikation voraus [Kaib02, S. 28f.], ein optimaler Kopplungsgrad zwischen Applikationen nicht notwendigerweise. Das Wiederverwendungspotenzial steigt also mit zunehmender Entkopplung.

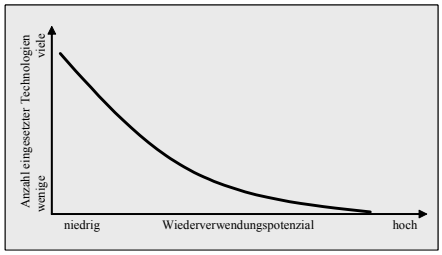

Abbildung 9

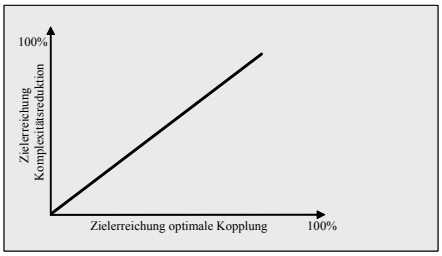

Abbildung 10

Wechselwirkungen zwischen optimaler Wiederverwendung und minimaler Infrastrukturkomplexität und -kosten (vgl. Abbildung 9)

Sind zu entwickelnde wieder verwendbare Komponenten unabhängig von der Infrastruktur, beeinflussen sich beide Ziele nicht. Sind sie jedoch abhängig, hat wieder die Anzahl eingesetzter Technologien Einfluss auf die Wiederverwendungsquote, da entschieden werden muss, für welche Technologie eine wieder verwendbare Komponente entwickelt werden soll (z.B. Entwicklung eines Corba-Service und/oder Entwicklung eines WebService). Eine hohe Technologievielfalt hat also einen negativen Einfluss auf die Wiederverwendung, sofern die Komponenten in Abhängigkeit der Infrastruktur entwickelt werden [Kaib02, S27ff.]. Auf der anderen Seite fördert der Einsatz weniger oder nur einer Technologie die Wiederverwendung, da keine Alternativen bei der Entwicklung neuer Komponenten wahrgenommen werden können.

Wechselwirkungen zwischen Komplexitätsreduktion und optimalem Kopplungsgrad (vgl. Abbildung 10)
Werden durch Desintegration viele Domänen gebildet, so hat dies viele lose gekoppelte Applikationen zur Folge, da zwischen Domänen lose, innerhalb von Domänen eng gekoppelt werden soll. Beide Ziele beeinflussen sich unmittelbar: Wird dem Ziel der Komplexitätsreduktion näher gekommen, wird auch dem optimalen Kopplungsgrad näher gekommen, sofern innerhalb von Domänen eng und domänenübergreifend lose gekoppelt wird.

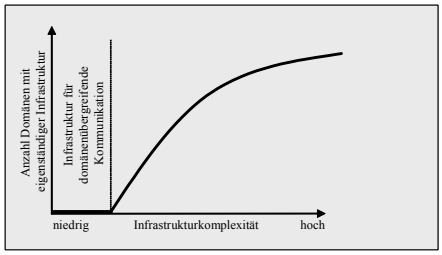

Abbildung 11

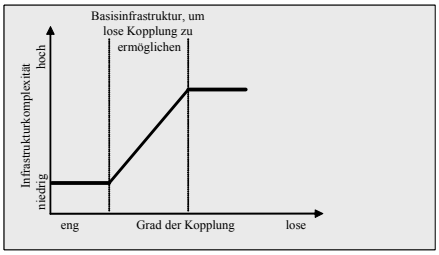

Abbildung 12

Wechselwirkungen zwischen Komplexitätsreduktion und minimaler Infrastrukturkomplexität und -kosten (vgl. Abbildung 11)

Wird die Infrastruktur unabhängig von gebildeten Domänen zur Komplexitätsreduktion zentral entwickelt und betrieben, sind beide Ziele zueinander indifferent. Werden jedoch innerhalb einzelner Domänen eigenständige Infrastrukturen aufgebaut, die unabhängig voneinander sind und nur zur domänenübergreifenden Integration eine zentrale Infrastruktur aufgebaut, steigen Gesamtinfrastrukturkomplexität und -kosten pro Domäne, in der autonome Infrastruktur aufgebaut und betrieben wird.

Wechselwirkungen zwischen optimalem Kopplungsgrad und minimaler Infrastrukturkomplexität und -kosten (vgl. Abbildung 12)

Bei ausschließlich enger Kopplung wären die Infrastrukturkosten gering, da keine zusätzlichen Infrastrukturkomponenten eingesetzt werden müssen, um die „Entfernung zwischen Applikationen zu überbrücken“ (beispielsweise zusätzliche Middleware zum Transport/Transformation von Nachrichten). Da aber nicht ausschließlich eng gekoppelt werden kann, müssen solche zusätzlichen Infrastrukturkomponenten vorhanden sein. Sind diese vorhanden und die Mechanismen der losen Kopplung können damit abgebildet werden, steigt die Infrastrukturkomplexität nicht mehr an.

6 Zusammenfassung und Ausblick

Auf Grundlage der Sammlung von Erfolgsfaktoren für die Applikationsintegration, die auf der Analyse von Publikationen und Praxisprojekten beruht, fokussierte dieser Beitrag auf die Diskussion möglicher Kennzahlen zur Messung der jeweiligen Zielerreichung und auf die Analyse der Zusammenhänge zwischen den Teilzielen. Beide Aspekte sind wichtig, um Hypothesen abzuleiten, die es nach empirischer Validierung ermöglichen, effektive Planungs- und Steuerungsmechanismen für die Applikationsarchitektur zu entwickeln. Diese wiederum sind unabdingbar, um die Applikationslandschaft mit dem Ziel der Annäherung an den „optimalen Integrationsgrad“ kontinuierlich weiterzuentwickeln.

Neben der Signifikanz der argumentativ abgeleiteten, in den Abbildungen 3 bis 12 illustrierten Zusammenhängen zwischen Zielen, ist auch der Zusammenhang dieser Ziele mit dem Agilitäts-Oberziel sowie die Vollständigkeit des betrachteten Zielsystems einer empirischen Validierung zu unterziehen. Zur Messung der Zielerreichung wurden Kennzahlen vorgeschlagen, die eine kontinuierliche Überwachung ermöglichen. Auch hier muss hinterfragt werden, inwieweit die definierten Kennzahlen vollständig sind und welchen quantitativen Einfluss einzelne Kennzahlen auf die Zielerreichung haben. Entsprechende Untersuchungen, für die der vorliegende Beitrag eine wichtige Grundlage bildet, werden zurzeit durchgeführt.

Literatur

[AmMo04] Ambrose, C.; Morello, D.: Designing the Agile Organization: Design Principles and Practices. Working Paper, Gartner Group, 2004.

[Aust03] Aust, H.: Einführung von EAI bei der PostFinance. 12. St. Galler Anwenderforum, St. Gallen, 2003.

[Bath03] Bath, U.: Web Services als Teil einer serviceorientierten Architektur. EAI-Forum Schweiz 2003, Regensdorf, 2003.

[Boe+ 00] Boehm, B. W.; Abts, C.; Brown, A. W.; Chulani, S.; Clark, B. K.; Horowitz, E.; Madachy, R.; Reifer, D.; Steece, B.: Software Cost Estimation with Cocomo II. Prentice Hall: New Jersey, 2000.

[Cumm02] Cummins, F. A.: Enterprise Integration. John Wiley & Sons Inc.: New York, et al., 2002.

[Endr03] Endries, T.: Schenker AG – EAI. Integration Management Day St. Gallen, 27.05.2003.

[Fres95] Frese, E.: Grundlagen der Organisation. Konzepte – Prinzipien – Strukturen. 6. Auflage. Gabler: Wiesbaden, 1995.

[FrHe04] Fridgen, M.; Heinrich, B.: Investitionen in die unternehmensweite Anwendungssystemintegration – Der Einfluss der Kundenzentrierung auf die Gestaltung der Anwendungslandschaft. Working Paper, Universität Augsburg, 2004.

[Frie03] Friederich, M.: Zusammenspiel verschiedener Integrationstechnologien und -werkzeuge bei der Züricher Kantonalbank. EAI-Forum Schweiz 2003, Regensdorf, 2003.

[Grög03] Gröger, S.: Enterprise Application Integration in the Financial Services Industry. Integration Management Day St. Gallen, 27.05.2003.

[Haf+04] Hafner, M., Schelp, J., Winter, R.: Architekturmanagement als Basis effizienter und effektiver Produktion von IT-Services. Erscheint in: HMD - Praxis der Wirtschaftsinformatik, 41, Heft 237, 2004.

[Hauf89] Haufs, P.: DV Controlling: Konzeption eines operativen Instrumentariums aus Budgets, Verrechnungspreisen, Kennzahlen. Physica-Verlag: Heidelberg, 1989.

[Hofe03] Hofer, A.: Projekt SBB CUS: EAI ermöglicht eine Erhöhung der Kundeninformations-Qualität im öffentlichen Verkehr. 12. St. Galler Anwenderforum, 15.09.2003.

[Jäge01] Jäger-Goy, H.: Führungsinstrumente für das IV-Management. Dissertation, Johannes-Gutenberg-Universität Mainz, 2001.

[Kaib02] Kaib, M.: Enterprise Application Integration – Grundlagen, Integrationsprodukte, Anwendungsbeispiele. Zugleich Dissertation Universität Marburg, DUV: Wiesbaden, 2002.

[Knec03] Knecht, R.: Application Architecture Framework UBS-WMBB. Integration Management Day, St. Gallen, 28.05.2003.

[Kral03] Krallmann, H.: Transformation einer industriell geprägten Unternehmensstruktur zur einer service-orientierten Organisation. Symposium des Instituts für Wirtschaftsinformatik "Herausforderungen der Wirtschaftsinformatik in der Informationsgesellschaft", Universität Leipzig, 2003, S. 1-12.

[Krcm90] Krcmar, H.: Bedeutung und Ziele von Informationssystemarchitekturen. In: Wirtschaftsinformatik, 32, 1990, Nr. 5, S. 395-402.

[Krcm03] Krcmar, H.: Informationsmanagement. 3. Auflage, Springer: Berlin, 2003.

[KuSc03] Kuster, S.; Schneider, M.: Banking Bus – EAI-Plattform der Raiffeisengruppe Schweiz. Integration Management Day, St. Gallen, 27.05.2003.

[Lint00] Linthicum, D. S.: Enterprise Application Integration. Addison Wesley: Reading, Massachusetts, 2000.

[Lisk03] Liske, C.: Advanced Supply Chain Collaboration enabled bei EAI. 12. St. Galler Anwenderforum, 15.09.2003.

[Malh96] Malhotra, Y.: Enterprise Architecture: An Overview, @BRINT Research Institute, http://www.brint.com/papers/enterarch.htm, 1996, Abruf am 2004-06-17.

[MaRo00] Martin, R.; Robertson, E.: A Formal Enterprise Architecture Framework to Support Multi-model Analysis. In Proceedings of the 5th CAiSE/IFIP8.1 international workshop on evaluation of modeling methods in systems analysis and design, Stockholm, 2000.

[McDa99] McDavid, D. W.: A standard for business architecture description. In: IBM Systems Journal, 38, 1999, Nr. 1, S. 12-31.

[Moll03] Moll, T.: Firmenübergreifendes EAI-Netzwerk – Integrierte Umsetzung von Geschäftsprozessen über einen Marktplatz als EAI-Hub. Integration Management Day, St. Gallen, 27.05.2003.

[Öst+92] Österle, H.; Brenner, W.; Hilbers, K.: Unternehmensführung und Informationssystem – Der Ansatz des St. Galler Informationssystem-Managements. Teubner: Stuttgart, 1992.

[Reic97] Reichmann, T.: Controlling mit Kennzahlen und Managementberichten: Grundlagen einer systemgestützten Controlling-Konzeption. 5. überarb. und erw. Auflage, Vahlen: München, 1997.

[Rose99] Rosemann, M.: Gegenstand und Aufgaben des Integrationsmanagements. In: Scheer, A.W.; Rosemann, M.; Schütte, R. (Hrsg.): Integrationsmanagement. Arbeitsbericht Nr. 65, Westfälische Wilhelms-Universität, Münster, 1999.

[Ruh+01] Ruh, W. A.; Maginnis, F. X.; Brown, W. J.: Enterprise Application Integration. John Wiley & Sons Inc.: New York et al., 2001.

[ScHa04] Schwinn, A.; Hagen, C.; Measured Integration – Metriken für die Integrationsarchitektur. Erscheint in: Schelp, J.; Winter, R. (Hrsg.): Integrationsmanagement. Springer: Berlin et al., 2004.

[ThIr01] Themistocleous, M.; Irani, Z.: Benchmarking the benefits and barriers of application integration. In: Benchmarking, 8, 2001, Nr. 4, S. 317-331.

[Wint03] Winter, R.: Modelle, Techniken und Werkzeuge im Business Engineering. In: Österle, H.; Winter, R. (Hrsg.): Business Engineering. 2. Aufl., Springer: Berlin et al., 2003, S. 87-118.

[Wint05] Winter, R.: Ein Modell zur Visualisierung der Anwendungslandschaft als Grundlage der Informationssystem-Architekturplanung. Erscheint in: Schelp, J.; Winter, R. (Hrsg.): Integrationsmanagement. Springer: Berlin et al., 2005.

[You+99] Youngs, R.; Redmond-Pyle, D.; Spass, P.; Kahan, E.: A standard for architecture description. In: IBM Systems Journal, 38, 1999, Nr. 1, S. 32-50.

[Zach87] Zachman, J. A.: A Framework for Information Systems Architecture. In: IBM Systems Journal, 26, 1987, Nr. 3, S. 276-292.

[Zach99] Zachman, J. A.: Enterprise Architecture: The Past and the Future. http://www.dmreview.com/article_sub.cfm?articleId=2187, 2000, Abruf am 2004-06-19.

[Zach01] Zachman, J. A.: You Can't 'Cost-Justify' Architecture. In: DataToKnowledge Newsletter (Business Rule Solutions LLC), 29, Nr. 3, 2001.

[Zaha00] Zahavi, R.: Enterprise Application Integration with CORBA. John Wiley & Sons Inc.: New York, et al., 2000.