IT-Dienstmodellierung

Modellierung von Beziehungen und Abhängigkeiten zwischen Geschäftsprozessmodellen, Softwarekomponenten und der IT-Infrastruktur

Norbert Diernhofer

Lehrstuhl Prof. Dr. Dr. h.c. Manfred Broy
der Technischen Universität München

IT-Dienstmodellierung

Modellierung von Beziehungen und Abhängigkeiten zwischen Geschäftsprozessmodellen, Softwarekomponenten und der IT-Infrastruktur

Norbert Diernhofer

Vollständiger Abdruck der von der Fakultät für Informatik
der Technischen Universität München zur Erlangung des akademischen Grades eines
Doktors der Naturwissenschaften (Dr. rer. nat.)
genehmigten Dissertation.

Vorsitzender:	Univ.-Prof. Dr. Helmut Krcmar
Prüfer der Dissertation:	
	1. Univ.-Prof. Dr. Dr. h. c. Manfred Broy
	2. Univ.-Prof. Dr. Martin Bichler

Die Dissertation wurde am 19.12.2006 bei der Technischen Universität München eingereicht und durch die Fakultät für Informatik am 16.04.2007 angenommen.

Zusammenfassung (1)

Ausgangssituation

Der Betrieb immer komplexerer IT-Systeme, die miteinander interagieren und die externe Systeme an Unternehmen anbinden, ist von einer starken Dynamik geprägt. Die Integration zwischen verschiedenen Systemen nimmt dabei immer mehr zu, Ansätze wie Service-orientierte Architekturen und die einfache Integration dieser Services auf der Basis von Webservices in Büroanwendungen wie Microsoft Office, die auch von Endanwendern durchgeführt werden können, machen die Planung und Verwaltung einer IT-Landschaft schwieriger, da oft überhaupt keine zentrale Planung mehr erfolgen kann. Das Fehlen eines Modells zur Beschreibung der Abhängigkeiten der Dienste voneinander und der IT-Systeme, auf denen sie ausgeführt werden, kann zu einem undurchsichtigem Geflecht in einer IT-Landschaft führen. Darüber hinaus zeichnen sich momentan weitere Entwicklungen ab, die diese Situation verschärfen:

- Der zunehmende Einsatz von Virtualisierungslösungen, bei denen auf einem Server mehrere virtuelle Maschinen ausgeführt werden, die physikalische Server emulieren, um die Auslastung der einzelnen physikalischen Maschinen und der Gesamtinfrastruktur zur steigern
- Die Notwendigkeit, die vorhandenen IT-Systeme vor immer ausgeklügelteren und spezifischeren Angriffen auf einzelne Dienste besser zu schützen und Sicherheitsmechanismen zu integrieren, die weit über die Möglichkeiten einen herkömmlichen, portbasierten Firewall hinausgehen. Dies führt zu noch komplexeren Strukturen und benötigt gleichzeitig ein wesentlich genaueres Wissen über die Struktur und die gegenseitigen Abhängigkeiten der zu schützenden Anwendungen, um entsprechende Maßnahmen etablieren und ihre Funktion bewerten zu können.

Die Möglichkeit der Beschreibung dieser IT-Dienste trägt zu einer Verbesserung dieser Situation bei und erleichtert nicht nur das Management der IT-Dienste auf einer technischen Ebene, sondern kann auch durch rechtliche Rahmenbedingungen wie die Forderung nach einer durchgängigen Dokumentation der Prozesse eines Unternehmens notwendig sein. Dies gilt gerade dann, wenn ein Unternehmen die etablierten IT-Prozesse nach ISO 20.000 zertifizieren möchte oder die Einhaltung rechtlicher Rahmenbedingungen wie SOX oder HIPAA auch für die IT-Infrastruktur belegt werden sollen oder müssen.

Der Beitrag der Arbeit

Wir stellen ein Modell vor, das die Modellierung der Abhängigkeiten zwischen verschiedenen IT-Diensten ermöglicht. Dabei betrachten wir sowohl horizontale Abhängigkeiten der verschiedenen Dienste einer Ebene, als auch die vertikalen Abhängigkeiten zwischen den Ebenen IT-Infrastruktur, Software und Geschäftsprozessen. Durch das Funktionsmodell, das eine zusätzliche Ebene der Beschreibung der realisierten Funktionen einführt, werden die Modelle besser strukturiert. Auf den verschiedenen Ebenen des Modells werden Teilmodelle angegeben, bzw. Informationen beschrieben, die bereits existierende Modelle enthalten müssen, um in das Gesamtmodell integriert werden zu können. Dabei werden sowohl die Softwarekomponenten als auch die IT-Systeme, auf denen diese ausgeführt werden, modelliert. Darüber hinaus lassen sich Geschäftsprozesse auf die IT-Systeme und Softwarekomponenten abbilden, die diese

Zusammenfassung (2)

realisieren. Unser Modell stellt den „Klebstoff“ zwischen den verschiedenen Ebenen dar, es beschreibt die Abhängigkeiten und die Beziehungen zwischen den Ebenen. Es ist kein allumfassendes Weltmodell, das alle Informationen eines IT-Dienstes im Sinne des Asset-Managements oder des Dienstmonitorings enthält.

Als Ausgangspunkt für die Modellierung verwendet das Modell die Schnittstellen zwischen den verschiedenen Systemen. Auf der Ebene der Softwarekomponenten sind dies angebotene Schnittstellen, wie Implementierungen von Facade-Pattern und die entsprechenden Aufrufe dieser Schnittstellen durch andere Softwarekomponenten. Auf der Ebene der IT-Infrastruktur werden diese Schnittstellen durch die Netzwerkinfrastruktur umgesetzt. Wir stellen in diesem Zusammenhang einen Prototyp vor, der die Generierung von Modellen der Abhängigkeiten von verschiedenen Softwarekomponenten voneinander aus dem Code einer Anwendung erlaubt. Damit ermöglichen wir die Wiederverwendung von Modellen und Ergebnissen der Softwareentwicklung und fördern so die Verschränkung von Entwicklung und Betrieb von IT-Diensten. Dadurch werden Medienbrüche zwischen diesen beiden Phasen des Lebenszyklus eines IT-Dienstes vermieden und eine iterative Weiterentwicklung der Dienste durch die Verwendung von Informationen und Modellen aus dem Betrieb erleichtert.

Diese starke Orientierung an der technischen Realisierung eines IT-Dienstes führt zu Modellen, die wesentlich näher an der IT-Infrastruktur angebunden sind und wesentlich genauere Aussagen zulassen als existierende Ansätze, die sich mehr auf einer strategischen Ebene bewegen. Beispiele dafür sind ARIS oder die Modelle der Softwarekarografie, wir stellen unser Modell im Laufe der Arbeit diesen Modellen gegenüber. Durch die Modellierung von Beziehungen wie Schnittstellenhierarchien werden hier technisch komplexe Abläufe auf eine sehr einfach zu verstehende und zu verwendende Basis abgebildet. Ein weiteres wichtiges Mittel der Modellierung, vor allem auf der Ebene der Software und der IT-Infrastruktur, sind Containerbeziehungen, zum Beispiel Laufzeitumgebungen wie .NET oder J2EE, oder die schon beschriebenen Beziehungen innerhalb einer virtualisierten IT-Infrastruktur. Auch diese Beziehungen lassen sich im Modell abbilden. Auf der Ebene der IT-Systeme sind die Abhängigkeiten zwischen den Serversystemen durch die Kommunikationskomponenten und die Kommunikationskanäle gegeben. Diese Umfassen neben den Netzwerkkomponenten auch die physikalische Verkabelung der Infrastruktur. Durch die Abbildung der Geschäftsprozesse auf Softwarekomponenten und die Beschreibung der Verteilung der Softwarekomponenten auf den IT-Systemen ergibt sich ein durchgängiges Modell, das von den Geschäftsprozessen bis zu den IT-Systemen reicht und diese miteinander verbindet. Das Funktionsmodell beschreibt die realisierten Funktionen der verschiedenen Ebenen. Es ist vor allem bei komplexen Modellen eine wichtige Hilfe zur besseren Strukturierung der Teilmodelle und erleichtert den Vergleich von verschiedenen Realisierungen einer Funktion.

Weitere Anwendungsmöglichkeiten

Neben der Modellierung der Abhängigkeiten und Abläufe geben wir auch Möglichkeiten zur Modellierung von Ausfällen und ihren Auswirkungen an. Damit kann das Modell zum Impact Management verwendet werden und mit dem Modell die Auswirkung von Ausfällen auf die Gesamtinfrastruktur simuliert werden. Darüber hinaus liefert das Modell auf Basis dieser Funktionen Informationen, die das Change Management unterstützen können. So können Veränderun-

gen in der Infrastruktur simuliert werden, bevor diese real durchgeführt werden. Somit lassen sich aber auch die Veränderungen von Service Level Agreements und die daraus resultierenden geänderten Anforderungen an die IT-Infrastruktur analysieren. Das Modell kann so als Basis für ein Tool zur Automatisierung von IT-Infrastrukturen dienen. Verändern sich die Service Level Agreements eines Dienstes, wie Anforderungen an die Reaktionszeit oder die Zahl der gleichzeitigen Benutzer, so kann ein entsprechendes Tool durch das Hinzuschalten zusätzlicher Kapazitäten in der Infrastruktur diese Änderungen entsprechend umsetzen.

Das Modell kann außerdem dazu verwendet werden, die Sicherheit der Systeme zu überprüfen. So lassen sich die vorhandenen Kanäle zwischen den Systemen mit dem Modell analysieren, nicht notwendige Verbindungen können getrennt und die Abschottung verbessert werden. Darüber hinaus wird deutlich, welche Verbindungen von welchen Diensten benutzt werden. Im Falle eines Angriffs wird es so möglich, die Auswirkungen auf das Unternehmen abzuschätzen und zu entscheiden, ob der potentiell zu erwartende Schaden durch den Angriff größer ist als der Schaden, der bei der Abschaltung einer speziellen Komponente entsteht, wenn der dadurch realisierte Geschäftsprozess nicht mehr ausgeführt werden kann.

Die modellierten Funktionen können darüber hinaus zur Bewertung der Ausfall- und Datensicherheit genutzt werden. Dadurch kann das Modell einen Beitrag bei der Etablierung eines umfassenden Risikomanagements leisten, was von einigen rechtlichen Rahmenwerken mittlerweile gefordert wird. Im Gegensatz zu einem Ansatz, der diese Vorgaben auf den einzelnen Ebenen isoliert umsetzt, können hier auch für einzelne Dienste unterschiedliche Umsetzungsstrategien gewählt werden, da die IT-Systeme, die den Dienst erbringen, beschrieben sind. So lassen sich auf diese isoliert strengere oder schwächere Vorgaben für den Betrieb umsetzen.

Abgrenzung

Im Gegensatz zu den Herangehensweisen anderer, mehr auf strategische Planung und an den Geschäftsprozessen orientierten Vorgehensweisen wie der Softwarekartographie oder ARIS liegt der Fokus dieser Arbeit auf einem integrierten Bild, das die Ebenen der Infrastruktur bis zu den Geschäftsprozessen mit ihren jeweiligen Modellen und Details einschließt und ihre Abhängigkeiten und Beziehungen beschreibt. Die Softwareebene wird dabei wie die Infrastruktur bewusst nicht grobgranular dargestellt, sondern detailliert mit all ihren Komponenten modelliert. Dadurch ergibt sich ein vollständiges Bild, das auch einzelne Teile der Ebenen wie Subnetze in der Infrastruktur erfasst. Nur mit diesem Grad an Detaillierung ist es möglich, operative Entscheidungen wie die Platzierung von Servern in verschiedenen Subnetzen zu überprüfen und zu planen, da nur dadurch die notwendigen Informationen zur Vorbereitung der Entscheidung vorhanden sind.

Es entsteht aber auch kein Weltmodell, das alle Informationen über einen IT-Dienst im Sinne eines Asset-Managements oder eines Monitoring-Tools enthält. Entsprechende Ansätze wie die durch ITIL beschriebenen CMDBs verfolgen einen anderen Ansatz, dem wir unser Modell gegenüberstellen. Der wichtigste Unterschied ist dabei, dass sich unser Modell vor allem auf die Modellierung der Beziehungen und Abhängigkeiten konzentriert.

Zusammenfassung (4)

Die Einbeziehung von Ergebnissen der Softwareentwicklung erleichtert die Erstellung von Modellen und hilft, dem Informationsverlust zwischen Entwickler und Betreiber vorzubeugen. Auch in diesem Bereich zeigen die anderen Techniken, dass sie einen anderen Ansatz verfolgen.

Integration anderer Modelle

Durch die sehr kompakten Teilmodelle ist eine Integration anderer Modellierungstechniken einfach möglich, da die Erstellung einer Abbildung zwischen verschiedenen Modellierungstechniken und unserem Modell vereinfacht wird. Vorhandene Modelle können durch diese Abbildung in ein IT-Dienstmodell integriert werden. Gerade auf der Ebene der Softwarekomponenten und der Geschäftsprozesse gibt eine Vielzahl an existierenden Modellierungstechniken. Wir beschreiben ein Beispiel, wie sich ARIS als Teilmodell für Geschäftsprozesse in unser Gesamtmodell integrieren lässt. Damit wird es möglich, bereits existierende Modelle auf den verschiedenen Ebenen in ein Gesamtmodell zu integrieren.

Wichtig ist in diesem Zusammenhang, dass eventuell schon bei der Entwicklung eines Dienstes ein umfassendes Modell für diesen speziellen Dienst entworfen wird. Relevant für die IT-Infrastruktur und die IT-Dienste eines Unternehmens ist aber nicht nur die Modellierung eines einzigen IT-Dienstes, sondern aller IT-Dienste eines Unternehmens. Bei dieser Herangehensweise kann es darüber hinaus vorkommen, dass in verschiedenen Projekten auf derselben Ebene eines IT-Dienstes verschiedene Modellierungstechniken verwendet werden. Unser Modell kann durch die Integration dieser verschiedenen Modelle auch zur Vereinheitlichung und Konsolidierung des Gesamtmodells beitragen.

Sind auf den verschiedenen Ebenen noch keine Modelle vorhanden, ermöglicht unser Modell eine sehr einfache und schnelle Modellierung der jeweiligen Ebene, die auch eine spätere Erweiterung auf eine größere und mächtigere Modellierungstechnik auf dieser Ebene erlaubt, ohne die Informationen über die Beziehungen zu verlieren. Damit wird durch unser IT-Dienstmodell der Einstieg in die Modellierung der IT-Dienste eines Unternehmens vereinfacht.

Wir beschreiben mit einem Prototyp, mit dem sich Regeln für eine Firewall erzeugen lassen, wie Informationen aus dem Quellcode einer Anwendung für den Betrieb generiert werden können. Dazu verwendet dieser Prototyp die Informationen aus der Softwarenentwicklung, überführt sie in eine angepasste Version unseres Modells und generiert aus diesen Informationen einen Regelsatz für eine Firewall. Dieses Beispiel illustriert, wie durch unser Modell die Entwicklung und der Betrieb von Software miteinander verschränkt werden kann.

Anpassbarkeit / Tailoring

Das Modell kann in verschiedenen Detaillierungsstufen und in Teilen eingesetzt werden. Wir geben entsprechende Ansatzpunkte für ein Tailoring des Modells an und beschreiben verschiedene Ausprägungsstufen. Damit wird es möglich, das Modell an die jeweiligen Bedürfnisse für den Einsatz anzupassen. Dies erleichtert vor allem den Einstieg in die Modellierung von IT-Diensten. Darüber hinaus kann gerade bei kleinen Anwendungssystemen oder Unternehmen, die heute keine Modelle ihrer Anwendungssysteme haben, so ein Einstieg in die Modellierung erfolgen und die Modelle schrittweise in ihrem Umfang und ihrer Granularität angepasst wer-

den. Durch die Verwendung der Tailloringvorgaben ist darüber hinaus sichergestellt, dass sich Basismodelle anschließend auf den vollen Umfang des Modells erweitern lassen.

Aufbau der Arbeit

Die Arbeit stellt das Gesamt- und die Teilmodelle vor. Wir gehen zuerst auf die verschiedenen Ebenen eines IT-Dienstes ein, beschreiben das Szenario und leiten daraus die Anforderungen an unser Modell ab. Dabei gehen wir auch auf existierende Techniken und Modelle ein. Anschließend beschreiben wir das Modell, das auf der Basis der Graphentheorie entwickelt wurde. Wir geben die verschiedenen Elemente und Bestandteile der Teilmodelle an und beschreiben die Funktionen, die zur Erstellung einer konkreten Instanz verwendet werden. Diese Funktionen stellen unter anderem die Konsistenz der Instanzen des Modells sicher.

Durch die Verwendung eines einfachen Basismodells wie Graphen ist das Modell leicht verständlich und kann auch in der täglichen Kommunikation oder mit Stift und Papier verwendet werden. Das Modell beschreibt dabei eine Denkweise, wie komplexe IT-Dienste verstanden und beschrieben werden können. Es orientiert sich dabei sehr stark an der Denkweise seiner Nutzer, die gedanklich vor sich Softwaresysteme, Geschäftsprozesse oder IT-Systeme sehen und diese beschreiben. Durch das Modell wird ein gemeinsames Verständnis etabliert, das auf einer formalen Basis steht. Durch die durchgängige Modellierung wird außerdem ein Verständnis für die speziellen Anforderungen der verschiedenen Ebenen geschaffen. Dabei werden die speziellen Anforderungen und Einschränkungen, wie Subnetze auf der Ebene der IT-Infrastruktur, integriert und automatisch berücksichtigt. Die Integration von Modellen der Softwareentwicklung ermöglicht eine durchgängige Beschreibung von IT-Diensten und deren Beziehungen während ihres gesamten Lebenszyklus und erleichtert so die Inbetriebnahme und das Management der verwendeten Softwarekomponenten.

Die Abstützung auf die Graphentheorie als Basismodell sowie die festgelegten Funktionen zur Erstellung konkreter Modelle und zur Konsistenzsicherung ermöglichen aber auch, das Modell direkt in eine konkrete Applikation umzusetzen oder Teile des Modells in bestehende Werkzeuge zu integrieren. Damit stellt die Arbeit auch die Vorarbeit für die Erstellung eines entsprechenden Werkzeugs dar. Wir stellen in der Arbeit einen ersten Prototyp dazu vor, der die Modellierung von IT-Diensten grafisch mit Hilfe eines Editors ermöglicht.

Eine Evaluierung der Arbeit an einem konkreten Geschäftsfall bei Siemens Business Services (SBS) rundet die Arbeit ab. Dabei gehen wir vor allem auch die Integrierbarkeit von existierenden Modellen (ARIS) ein und beschreiben den durch die Erstellung des Modells erzielten Mehrwert und die Erfahrungen dieser Evaluierung. Ein Ausblick auf weitere Anwendungsszenarien für das Modell und mögliche Forschungsarbeiten schließt die Arbeit ab.

Inhalt (1)

Inhalt (2)

Abbildungen (1)

Abbildungen (2)

Abbildungen (3)

Definitionen (1)

Definitionen (2)

Funktionen (1)

Funktionen (2)

Funktionen (3)

Danksagung

Ich möchte mich an dieser Stelle bei Prof. Broy bedanken, dass er diese Arbeit ermöglicht hat. Die Entwicklung dieser Arbeit am Lehrstuhl war sehr spannend und interessant, ich konnte in den letzten Jahren sehr viel lernen und hatte dabei auch immer den Freiraum, eigene Ideen zu verwirklichen. Herzlichen Dank auch an Prof. Bichler für die Übernahme des zweiten Gutachtens.

Danken möchte ich auch Dr. Hans-Georg Völk und seinen Kollegen bei Siemens Business Services, für die Freiheit zu forschen, die tatkräftige Unterstützung, aber auch für die sehr konstruktiven Gespräche, die auf dem Weg zu diesem Modell sehr geholfen haben.

Meinen Kollegen am Lehrstuhl und am Institut, vor allem Franz Huber, Christoph Erdle, Marco Kuhrmann, Thomas Schneider, Katharina Spies, André Wittenburg und der ganzen IT-Abteilung gilt mein herzlicher Dank für sehr fruchtbare Diskussionen, Feedback und die Unterstützung in der täglichen Arbeit.

Großer Dank vor allem an meine Familie, Theresia, Melanie, Erhard und Andrea, für ihr Verständnis, Geduld, moralische Unterstützung und die Gespräche, die auf dem Weg zu dieser Arbeit sehr viel beigetragen haben. DANKE!

1 Einleitung

In diesem Kapitel gehen wir auf die aktuellen Rahmenbedingungen für die Erbringung von IT-Diensten in Rechenzentren von Unternehmen ein. Wir beschreiben dabei vor allem auch die Veränderungen, die sich aus den aktuellen Entwicklungen ergeben, sowie die daraus resultierenden Herausforderungen für das Management von IT-Diensten. Wir stellen das Ziel dieser Arbeit, die Verbesserung des Managements dieser Dienste vor und gehen auf die Vision, die hinter dieser Arbeit steckt, ein. Darüber hinaus gehen wir auf die Zielgruppe und unser Vorgehen ein. Wir beschreiben die Struktur unseres IT-Dienstmodells und gehen auf mögliche Erweiterungen ein. Abschließend stellen wir den Aufbau der Arbeit vor.

1.1 Aktuelle Rahmenbedingungen für IT-Dienste

Heute verwendete, gewachsene Softwarelandschaften zeichnen sich durch ihre gegenseitige Abhängigkeit aus. Durch neue Paradigmen wie Service-orientierte Architekturen (SOA) und die immer weiter voranschreitende Vernetzung zwischen verschiedenen IT-Systemen, wird dieser Aspekt verstärkt. Die Zusammenarbeit verschiedener Unternehmen, die Verlängerung von Zulieferketten sowie neue Unternehmensformen wie die „grenzenlose Unternehmung“, die Picot und Reichwald schon in [GU98] beschreiben, stellen neue Anforderungen an die IT-Infrastruktur, die sich dort in neuen Architekturen und Techniken wie Service-orientierten Architekturen, dem verstärkten Einsatz der Virtualisierung zur Verbesserung der Kostenstruktur und neuen Architekturen wie Grids niederschlagen. Die früher existierenden harten Grenzen zwischen verschiedenen IT-Systemen und Rechenzentren verschwinden dabei immer mehr. Die IBM CEO Studie [IBM06], bei der weltweit über 700 CEOs befragt wurden belegt dabei, dass diese Entwicklung keineswegs abgeschlossen ist. CEOs sehen heute vielmehr die Notwendigkeit, die IT-Infrastruktur und die Geschäftsprozesse als eine Einheit zu betrachten, um Innovationen in den Geschäftsprozessen voranzutreiben und so Wettbewerbsvorteile zu erarbeiten. Wichtig ist dabei, dass die IT-Infrastruktur flexibel genug ist, diese Veränderungen zu realisieren. Dabei spielt auch die Integration von Zulieferern und Partnern eine wichtige Rolle. Durch die Bildung von Partnerschaften können Unternehmen innerhalb ihrer Branche innovative Lösungen anbieten, oder in neue Märkte expandieren.

Diese Entwicklungen und Veränderungen der Unternehmensorganisation sorgen dafür, dass Abhängigkeiten der IT-Systeme auch zwischen verschiedenen Unternehmen auftreten, die keine gemeinsame IT-Plattform betreiben. Die Integration anderer Unternehmen in so entscheidende IT-Systeme wie die Produktionsprozesssteuerung sorgt dafür, dass Abhängigkeiten von Fremdsystemen entstehen, die sich außerhalb der Kontrolle der eigenen IT-Abteilung befinden. Dadurch entstehen in modernen Rechenzentren für die einzelnen Unternehmen immer komplexere IT-Landschaften, die zur Wertschöpfung maßgeblich beitragen und von immer größerer Bedeutung sind. Daraus ergeben sich auch entsprechend höhere Anforderungen an die Verfügbarkeit und die Sicherheit der Systeme.

Ein weiterer wichtiger Faktor ist die Flexibilität der Systeme im Bezug auf Veränderungen, die Möglichkeit zur Integration von anderen Systemen oder von neuen Partnern. Diese Flexibilität kann gerade bei Unternehmen mit sehr dynamischen Geschäftsbeziehungen und der daraus resultierenden häufig notwendigen Anpassung der IT-Infrastruktur an wechselnde Geschäftspartner für die Überlebensfähigkeit eines Unternehmens entscheidend sein. Dies stellt hohe Anforderungen an das Management der IT-Systeme und IT-Dienste des Unternehmens, für die eine ganzheitliche Sicht unabdingbar ist.

Die verstärkte wirtschaftliche Betrachtung von IT-Diensten und der IT-Infrastruktur in Unternehmen, die gerade in den letzten Jahren ausgehend von der Betrachtung der Total Cost of Ownership (TCO) bis zum verstärkten Outsourcing von IT-Dienstleistungen deutlich zugenommen hat, führt darüber hinaus dazu, dass die momentan vorhandenen „Silolandschaften“, bei denen für jede Aufgabe ein oder mehrere Server installiert wurden, auf denen isoliert die jeweilige Software ausgeführt wird, abgelöst werden von virtualisierten Plattformen, um die Auslastung der einzelnen Systeme zu steigern. Die Virtualisierung erfolgt dabei heute meist auf Basis von Softwarelösungen für virtuelle Maschinen, bei denen mehrere virtuelle Server auf einem

physikalischen Rechner emuliert werden. Zum einen lassen sich so Legacy Applikationen, die auf ein bestimmtes Betriebsystem angewiesen sind, auf aktueller Hardware weiterbenutzen, da in der virtuellen Maschine Hardwarekomponenten emuliert werden, die auch mit alten Betriebsystemversionen kompatibel sind, auf der anderen Seite steigt dadurch aber auch die Gesamtauslastung des physikalischen Servers, was zur Verbesserung der Kostenstruktur[1] beiträgt. Durch diese Technik lassen sich vor allem vorhandene Systeme relativ leicht ablösen, die meisten Hersteller entsprechender Produkte zur Virtualisierung bieten dazu Lösungen an[2]. Die Ausführung mehrerer virtueller Maschinen innerhalb eines physikalischen Servers steigert die Komplexität der Landschaften zusätzlich und generiert neue Abhängigkeiten der Systeme untereinander. Betrachtet man an diesem Punkt neue Entwicklungen, wie die von Intel[3] im Rahmen von VT-d vorgestellt Virtualisierung von Netzwerkschnittstellen oder die Initiativen von HP und CISCO, im Umfeld von Netzwerkswitches einen Sicherheitslayer einzuführen, der Benutzeranmeldung und später den Zustand des Rechners[4] in die Entscheidungen des Switches und die Konfiguration von VLAN's mit einbezieht, so entstehen hier zusätzliche Ebenen in der IT-Infrastruktur, die die Komplexität weiter steigern und dazu führen, dass sich Kommunikation nicht mehr nur auf die physikalische Verbindung zwischen verschiedenen IT-Systemen zurückführen lässt. Wichtig für die Entscheidung, ob eine Verbindung zwischen zwei Systemen vorhanden ist, ist vielmehr die Konfiguration und die Einflüsse von Systemen wie Virtualisierungslösungen oder dem Betriebsystem in einem Switch. Am Beispiel der beschriebenen HP Sicherheitslösung kann dies bedeuten, dass die Verbindung davon abhängt, ob ein gültiger Benutzer am System angemeldet ist, oder der Rechner alle Sicherheitsupdates installiert hat. Je nach dem wird der entsprechende Port am Switch in das eine oder andere VLAN geschaltet und so die Struktur des Netzwerkes geändert. Erweitere Funktionen erlauben es, dass die Entscheidung, ob eine Verbindung möglich ist oder nicht, im Falle einer erkannten Bedrohung für das Netzwerk unterschiedlich ausfallen können. All diese Einflüsse finden sich erst auf einer logischen Ebene und nicht auf der Ebene der physikalischen Verbindung. Die Systeme, die diese Logik realisieren, haben auf den ersten Blick auch keine direkte Verbindung mit den beiden Systemen. Die Betrachtung dieser Systeme, sowie die möglichen Konfigurationen, müssen jedoch bei der Betrachtung eines IT-Dienstes berücksichtigt werden.

Heute fehlt ein Modell zur Beschreibung dieser Abhängigkeiten, um die Einflüsse von Ausfällen, Angriffen oder Sicherheitsmaßnahmen, bzw. Veränderungen in der Struktur auf der technischen Ebene übergreifend darstellen und prüfen zu können. Auch Informationen über die durch die verschiedenen Softwaresysteme ausgeführten Geschäftsprozesse sind im Betrieb meist nicht zugänglich. Eine Abschätzung der Einflüsse von Störungen eines IT-System auf die Geschäftsprozesslandschaft eines Unternehmens ist deswegen meist nur schwer oder gar nicht möglich, da ein vollständiges Bild des IT-Dienstes fehlt. Dies erschwert sowohl das Management als auch die Weiterentwicklung von IT-Diensten und wirkt einer flexiblen IT-Infrastruktur und flexiblen Diensten entgegen.

Rüdiger Zarnekow stellt in [Zan05] einen weiteren wichtigen Aspekt vor, dem durch den Einsatz unseres Modells begegnet werden kann. Neue Rechtliche Rahmenbedingungen wie Sarbanes

---

[1] Die typische Auslastung entsprechender Server liegt nach Studien unterschiedlicher Analysten zwischen 5%-20%. Ähnliche Zahlen ergaben sich bereits vor Jahren im Bereich Direct attached Storage, also Speichersystemen, die direkt mit nur einem Server verbunden sind. Hier konnten durch Virtualisierungslösungen im SAN Steigerungen auf durchschnittlich 70-80% erreicht werden. Ähnliche Effekte erhofft man sich nun beim Einsatz dieser Konzepte im Bereich der Serversysteme und deren Virtualisierung.

[2] Die Lösungen wie VMware Converter oder Microsoft Virtual Server Migration Toolkit (VSMT) sind in der Lage, eine vorhandene Installation auf einem physikalischen Server in eine virtuelle Maschine zu transferieren. Dabei werden alle Einstellungen und alle Daten des Servers in die virtuelle Instanz übertragen. Eine Neuinstallation des Servers kann so entfallen.

[3] AMD bietet mittlerweile eine vergleichbare Technik an, die auch von aktuellen Virtualisierungslösungen unterstützt wird.

[4] Windows Vista/2008 Server Network Access Protection, CISCO Network Admission Control

Oxley (SOX), der Health Insurance Portability and Accountability Act (HIPAA), Basel II [Basel2] oder Corporate Governance Vorgaben stellen neue Anforderungen an das Management eines Unternehmens. Diese Vorgaben schlagen sich auch im Management von IT-Diensten nieder, da die Vorgaben durch diese neuen Rahmenbedingungen auf allen Unternehmensebenen umgesetzt werden müssen. Eine ganzheitliche Sicht der IT-Dienste ist also nicht nur notwendig, um diese besser verwalten zu können, sie ist die Grundlage, um IT-Dienste transparent zu machen und ein entsprechendes alle Aspekte des Unternehmens umfassendes Risikomanagement (Basel II) zu etablieren. Um Vorgaben und Empfehlungen, wie beispielsweise im BITKOM „Leitfaden E-Mail-Archivierung“ [BITKOM05] umsetzen zu können, müssen die verschiedenen Infrastrukturkomponenten und ihr Zusammenspiel analysierbar sein.

Notwendig ist dazu unter anderem die Etablierung angemessener IT-Managementprozesse, wie diese ITIL [ITIL] zum Beispiel vorgibt. Symons [Sym05] stellt dabei zurecht klar, dass IT-Governance und IT-Management durch die Anwendung von Standards wie BS15.000, das ITIL[5] und CobiT[6] zu einem Standard zusammenfasst, zwar einfacher möglich wird und diese eine gute Grundlage für den Aufbau eines eigenen IT-Governance Prozesses darstellen, diese Standards aber an die jeweilige Situation der IT-Dienste angepasst werden müssen. Auch dafür ist ein umfassendes Wissen der im Unternehmen eingesetzten Softwarekomponenten und der IT-Infrastruktur notwendig.

1.2 Struktur und Verantwortliche für einen IT-Dienst

Diese beschriebenen Herausforderungen an das Management von IT-Diensten beziehen sich vor allem auf die Softwareebene und die IT-Infrastruktur selbst, zu einem IT-Dienst gehört jedoch auch eine Beschreibung der Geschäftsprozesse und deren Beziehungen, die der Dienst auf der IT-Infrastruktur realisiert. Wir definieren in Kapitel 5.1.1 einen IT-Dienst formal, ein IT-Dienst im Sinne dieser Arbeit besteht vereinfacht aus Anwendungen, die Geschäftsprozesse implementieren und auf IT-Systemen ausgeführt werden.

Wir wollen in diesem Abschnitt eine Struktur für diese IT-Dienste und die entsprechenden Verantwortlichen kurz einführen, wir gehen auf die Anforderungen der verschiedenen Verantwortlichen in Kapitel 3.1 ein.

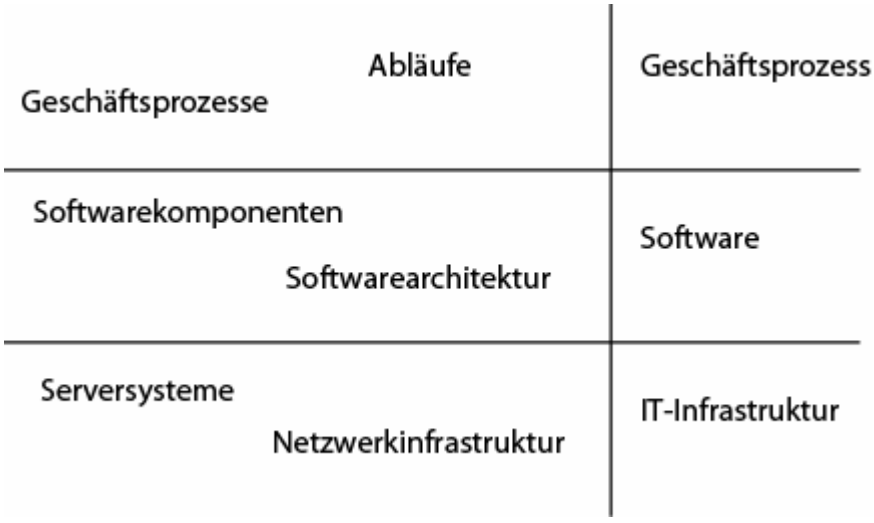

Abbildung 1 – Elemente eines IT-Dienstes[7]

[5] ITIL – IT Infrastructure Library – Modell zum Management von IT-Diensten

[6] CobiT – Control Objectives for Information and Related Technology – Standard für Kontrollziele zur Überwachung von IT-Diensten

[7] Alle Grafiken in der Arbeit sind, so keine anderen Quellen angegeben, eigene Grafiken

Die Organisation des Managements und der Verwaltung eines IT-Dienstes sind heute je nach Unternehmen und eventuell sogar je nach Dienst unterschiedlich organisiert. Bei Siemens SBS herrscht zum Beispiel meist eine Trennung entlang den Ebenen Entwickler, Administrator und Geschäftsprozessverantwortlicher vor [Kunze06]. Es sind jedoch auch andere Einteilungen möglich. Je nach Art der Organisation ergeben sich Vor- und Nachteile. Wir gehen im Weiteren von einer horizontalen Trennung der Verantwortlichkeiten aus, dabei sind Experten der jeweiligen Ebene für ihre Ebene verantwortlich.

Wir definieren drei Ebenen mit jeweils einem Verantwortlichen für die Ebene, wobei die Rollen zusammengefasst werden können:

- Geschäftsprozessebene
  Auf der Geschäftsprozessebene werden die Fachverfahren und die Abläufe in diesen Verfahren für einen Geschäftsprozess beschrieben. Diese Abläufe beinhalten die Schritte, die zur Ausführung eines Geschäftsprozesses notwendig sind. Der Geschäftsprozessverantwortliche ist fachlich für einen Geschäftsprozess zuständig. Er repräsentiert die Fachabteilung, die einen Geschäftsprozess verwendet oder verantwortet. Er definiert die fachlichen Anforderungen für die Realisierung eines IT-Dienstes.
- Softwareebene
  Auf der Softwareebene werden die Softwarekomponenten und die Architektur einer Anwendung beschrieben. Dabei sind auch Beziehungen zu Basisdiensten und Abhängigkeiten der Softwarekomponenten untereinander wichtig. Der Softwareentwickler implementiert oder adaptiert Software an die Bedürfnisse des Unternehmens und bringt so die fachlichen Anforderungen des Geschäftsprozessverantwortlichen in eine auf einem IT-System ausführbare Form. Die dafür eingesetzte Software muss dabei eventuell auch an die IT-Infrastruktur des Unternehmens angepasst werden.
- IT-Infrastruktur
  Die IT-Infrastruktur beschreibt alle Serversysteme, Speichergeräte und Netzwerkkomponenten, die die physikalische IT-Infrastruktur eines Unternehmens bilden. Verantwortlich für die IT-Infrastruktur ist dabei der Administrator. Er verantwortet die IT-Infrastruktur, installiert auf dieser Anwendungssysteme oder Softwarekomponenten und pflegt sie. Darüber hinaus ist er für die physikalische Infrastruktur und deren Wartung zuständig.

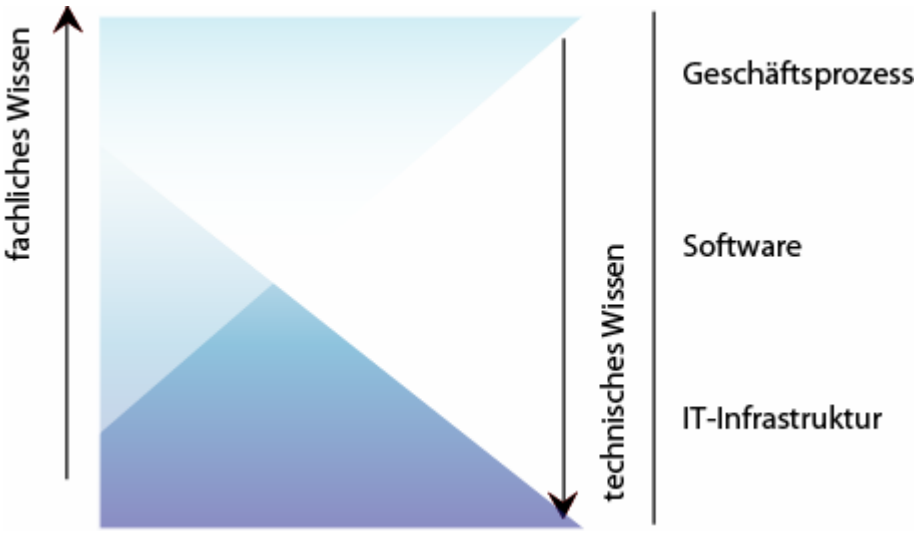

Abbildung 2 – Verschiedene Arten von Wissen über einen IT-Dienst

Für die unterschiedlichen Verantwortlichen ergeben sich verschiedene Perspektiven auf einen IT-Dienst. Die Verantwortlichen gehen jeweils von ihrer Ebene aus, die sie im Detail kennen. Wie bereits beschrieben, gehen wir von Experten auf der jeweiligen Ebene aus, die zusammen für einen IT-Dienst verantwortlich sind. Wir unterscheiden in diesem Fall zwischen fachlichem

und technischem Wissen über einen IT-Dienst. Durch die Beschreibung der Beziehungen zwischen den Ebenen durch unser Modell wird es ihnen so möglich, ein Verständnis für die jeweils relevanten Komponenten der anderen Ebenen zu entwickeln.

Die Abbildung 2 verdeutlicht diese zwei verschiedenen Arten von Wissen, die zusammengenommen das Gesamtwissen über einen IT-Dienst an sich und seine technische Erbringung liefern.

1.3 Ziel der Arbeit

Die beschriebene Entwicklung in den Rechenzentren führt zu einem immer verwobenerem Geflecht von Anwendungen und Systemen, die Struktur und die Beziehungen der Systeme untereinander werden dabei immer komplexer. Das Ziel dieser Arbeit ist es, die verschiedenen Ebenen eines IT-Dienstes zu modellieren, sowie die Abläufe in den Ebenen und die Abhängigkeiten zwischen den Ebenen zu beschreiben. Durch die Modellierung aller IT-Dienste eines Unternehmens entsteht so ein Modell, das alle IT-Dienste, ihre Ausführungsumgebung und ihre Beziehungen zueinander beschreibt.

Das in dieser Arbeit vorgestellt Modell ermöglicht dazu die Darstellung der Ebenen Geschäftsprozess, Softwarelandschaft, IT-Infrastruktur und die Modellierung der Bestandteile dieser Teilmodelle. Das Funktionsmodell beschreibt auf einer abstrakten Ebene die realisierten Features, strukturiert komplexe Modelle und ermöglicht den Vergleich verschiedener Implementierungen eines Geschäftsprozesses. Die Beschreibung von Auswirkungen von Ausfällen auf die einzelnen Komponenten und die Auswirkung auf die Abhängigkeiten der betroffenen Komponente werden durch entsprechende Funktionen in den jeweiligen Teilmodellen abgebildet.

Damit wird die Analyse der Abhängigkeiten auf den verschiedenen Ebenen vereinfacht und eine Möglichkeit aufgezeigt, das Geflecht der Anwendungen und der IT-Dienste zu beschreiben. Außerdem ist die Analyse der Systeme im Hinblick auf Fehlertoleranz und Sicherheit deutlich einfacher möglich, Methoden dafür werden in der Arbeit beschrieben. So können zum Beispiel so genannte „Single Points of failure“ identifiziert werden. Die Methoden können aber auch weiter verfeinert oder an andere Aufgaben angepasst werden. Das Modell kann so für den Administrator als eine Basis für ein ganzheitliches Management von IT-Diensten dienen. Wichtig ist dabei vor allem auch die Integration der verschiedenen Perspektiven der Verantwortlichen für einen IT-Dienst, was durch die verschiedenen Teilmodelle erreicht wird.

Das Modell ermöglicht es einem Geschäftsprozessverantwortlichen, ausgehend von seinen Geschäftsprozess die für ihn relevanten Softwarekomponenten und IT-Systeme zu identifizieren und sich so einen Überblick über die für ihn relevanten Teilkomponenten zu machen. Durch die Bereitstellung eines Modells, das die Abhängigkeiten und Beziehungen beschreibt, entsteht so eine gemeinsame Basis mit dem Softwareentwickler und dem Administrator, um Entscheidungen zur Veränderung oder Weiterentwicklung eines IT-Dienstes besser planen und organisieren zu können.

Gerade die Integration des Softwareentwicklers und der Softwareentwicklung selbst durch unser Modell stellt einen wichtigen Ansatz dar, der in anderen Arbeiten häufig nicht oder nur teilweise betrachtet wird. Unser Modell integriert den gesamten Softwarelebenszyklus und versucht, eine Brücke zwischen der Entwicklung von IT-Diensten und ihrer Erbringung zu schlagen. Durch die Verzahnung und die bessere Weitergabe von Informationen über die Applikation lassen sich Fehler bei der Inbetriebnahme vermeiden. Darüber hinaus entstehen so Modelle, die sowohl die Architektur der Anwendung als auch ihre konkrete Ausführungsumgebung enthalten. Durch die Vermeidung von Medienbrüchen können diese Modelle auch zwischen dem Betrieb und der Entwicklung direkt ausgetauscht werden. Dadurch können schon bei der Entwicklung der Applikation Einschränkungen, die für deren Betrieb gelten, mit berücksichtigt werden. Designentscheidungen, wie die Abstützung auf bereits existierende Basisdienste werden so deutlich einfacher möglich, da die Informationen über die Basisdienste in die Modelle integriert sind. Darüber hinaus stehen bei der Weiterentwicklung eines Dienstes Modelle und Informationen aus der Laufzeit eines Dienstes zur Verfügung, die gerade bei der Optimierung wichtige Informationen liefern können.

Für den Administrator wird es durch unser Modell möglich, die Softwarekomponenten auf der IT-Infrastruktur mit den Geschäftsprozessen zu verbinden und bei neuen Softwarekomponenten die Beziehungen zu existierenden Komponenten oder neue Abhängigkeiten zu erkennen. Dadurch wird die Planung bei der Einführung von neuen Anwendungen vereinfacht, weil sich bereits von Anfang an Probleme oder notwendige Änderungen an den existierenden Anwendungen abschätzen lassen. Im Falle des Ausfalls einer Komponente können durch das Modell die Auswirkungen auf andere Komponenten deutlich einfacher analysiert werden. Auch die sogenannte Root-Cause Analysis, bei der für einen Fehler in einer Komponenten das auslösende Ereignis gesucht wird, ist dadurch leichter möglich.

Die starke Integration der technischen Realisierung und die genau Modellierung der physikalischen Infrastruktur ist ein wichtiges Merkmal unseres Modells. Es befindet sich damit in der Mitte von zwei momentan verbreiteten Ansätzen in der Modellierung von IT-Diensten:

- Top-Down orientierte Ansätze wie die Softwarekartographie oder ARIS gehen von den Geschäftsprozessen aus und ermöglichen keine detaillierte Modellierung der technischen Infrastruktur. Damit ist die Bestimmung der Konsequenzen von Ausfällen nur schwer möglich, auch die Analyse technischer Abhängigkeiten verschiedener Dienste voneinander ist meist nicht oder nur schwer möglich.
- Bottom-Up orientierte Ansätze, die sich aus den Managementwerkzeugen der IT-Infrastruktur wie ITIL CMDBs entwickeln, haben heute noch keine Möglichkeit zur Integration einer Geschäftsprozesssicht auf die Dienste, da sie sich zu stark an den einzelnen Monitoring- und Konfigurationsaufgaben orientieren, aus denen sie entstanden sind. Für einen Verantwortlichen auf der Ebene der Geschäftsprozesse sind diese Tools und Modelle nicht verständlich und nicht anwendbar.

Unser Modell kann die Brücke zwischen diesen beiden Welten schlagen, indem es die Modelle der beiden Ansätze miteinander verbindet. Das fehlende Bindeglied dieser beiden Ansätze sind genau die Beziehungen zwischen den Ebenen.

1.4 Die Vision der Arbeit

Die weitereichende Vision dieser Arbeit ist ein Tool für IT-Governance oder ein IT-Service Leitstand[8], das das Management von IT-Diensten mit all ihren Bestandteilen auf fachlicher Ebene auf Basis von fachlichen Anforderungen ermöglicht. Dazu werden entsprechende Anforderungen wie garantierte Antwortzeit, Zahl der maximalen Nutzer oder aber beispielsweise die Durchlaufzeit einer Rechnung auf Ebene der Geschäftsprozesse definiert. Das Tool überwacht die gesamte Infrastruktur, alle damit verbundenen Softwarekomponenten und Dienste und kann diese entsprechend konfigurieren. Durch Übersetzung der fachlichen Anforderungen in die Konfiguration und Verteilung der konkreten Komponenten wird die Infrastruktur so angepasst, dass das Gesamtsystem die neuen Anforderungen erfüllt. Die im Tool zu implementierenden Übersetzungen zwischen den Ebenen, um fachliche Anforderungen in konkrete Arbeitsschritte auf der Hardware umzusetzen, ermöglichen dabei auch eine Abstraktion des Managements unterschiedlicher Softwaresysteme und Hardwarearchitekturen. Damit stellt das Tool primär kein Werkzeug für den Administrator dar, sondern gibt dem jeweiligen Verantwortlichen für den Geschäftsprozess alle Möglichkeiten zur Konfiguration seines Dienstes an die Hand. Dabei wird die konkrete Implementierung auf der Infrastruktur für ihn nicht sichtbar. Die heute noch manuellen Prozesse zur Umsetzung werden automatisiert. Ein entsprechendes Tool könnte auch alle Veränderungen an der Infrastruktur dokumentieren, bzw. Veränderungen mit regulatorischen Vorgaben überprüfen und Verstöße entsprechend melden. Darüber hinaus ist mit einem entsprechenden Tool auch in einer vollständigen virtualisierten Infrastruktur die Erkennung möglicherweise vorhandener Überkapazitäten, die noch nicht verkauft sind, erkennbar. Ein entsprechendes Werkzeug wäre auch interessant für den Vertrieb.

Diese Arbeit liefert die theoretischen Grundlagen für ein entsprechendes Tool. Dieses Tool kann damit zur stärkeren Automatisierung von Rechenzentren beitragen, man spricht in diesem

[8] Oft auch als Service Dashboard oder Service Cockpit bezeichnet

Zusammenhang auch von der Industrialisierung von IT-Infrastrukturen. Werden die Ziele der Arbeit vollständig erreicht, entsteht so eine Produktion von IT-Diensten, die vollautomatisch nur noch auf Basis der fachlichen Anforderungen der Geschäftsprozesse definiert werden kann. Das Tool realisiert dabei die Übersetzung zwischen den fachlichen Anforderungen und den notwenigen Aktionen und führt diese über entsprechende Managementtools aus. Wichtig ist hierbei, dass die verschiedenen Beteiligten Ebenen auch weiterhin integriert sind. Die Übersetzungsfunktionen müssen weiter vom Entwickler und Administrator angepasst und eingerichtet werden. Es steigt aber der Grad an Automatisierung. Interessant sind in diesem Zusammenhang sich heute abzeichnende und im wissenschaftlichen Umfeld bereits im Einsatz befindliche Grid-Infrastrukturen, die eine Basis für solche Rechenzentren sein könnten. In diesem Umfeld existieren jedoch noch viele Fragestellungen, die heute noch nicht gelöst sind. So haben die Hardwarearchitekturen einen Einfluss auf die Leistungsfähigkeit eines Servers, diese sind nicht direkt miteinander vergleichbar. Auch diese Unterschiede müsste ein entsprechendes Tool verstehen und in seine Berechnungen mit einbeziehen.

1.5 Modellstruktur

Unser Modell ermöglicht es, durch vier unterschiedliche Modellebenen mit jeweils eigenen Teilmodellen sowohl die IT-Infrastruktur, als auch die Softwarekomponenten, die damit realisierten Geschäftsprozesse und die Funktionen[9] der einzelnen Komponenten zu beschreiben. Sowohl die Abläufe innerhalb der Ebenen als auch die Beziehungen zwischen den Ebenen werden modelliert. Dadurch werden Informationen zwischen den Ebenen übertragbar, bzw. die Quelle für Anforderungen an eine Komponente erkennbar. Der Anforderung einer bestimmten maximalen Ausführungsdauer an eine Softwarekomponente und damit die Anforderungen an die Rechenleistung eines Servers kann so der dazu notwendige Geschäftsprozess zugeordnet werden. Bei der Zuordnung der Softwarekomponenten zu Infrastrukturkomponenten (Deployment) können diese Anforderungen integriert werden. Darüber hinaus wird es damit möglich, die Auswirkungen von Veränderungen dieser Anforderungen zu betrachten. So können in diesem konkreten Beispiel die Kosten für unterschiedliche Ausführungsdauern deutlich einfacher abgeschätzt werden, da alle beteiligten Komponenten identifizierbar und die Anforderungen auf Hardwarekosten umsetzbar sind.

Innerhalb der verschiedenen Teilmodelle sind alle für die Modellierung der Beziehung zwischen den Ebenen relevanten Elemente definiert[10].

- Geschäftsprozessmodell
  Auf der Ebene der Geschäftsprozesse sind die Geschäftsprozesse als Elemente definiert. Durch Verfeinerungsbeziehungen zwischen den Geschäftsprozessen werden diese strukturiert. Die Abläufe modellieren die Beziehungen zwischen den Geschäftsprozessen.
- Funktionsmodell
  Innerhalb des Funktionsmodells definieren wir Features, die verfeinert werden können. Diese Features beschreiben die Funktionen und Aufgaben von Elementen aus anderen Ebenen. Beziehungen oder Beschreibungen der Funktion einer Komponente werden durch die Modellierung einer Beziehung zwischen dem Element und dem jeweiligen Feature realisiert.

  Diese Ebene erlaubt es, komplexe Modelle besser zu strukturieren. Damit werden die konkreten Realisierungen den Features zugeordnet, die sie implementieren. Dies erleichtert zum einen die Navigation und die Arbeit mit komplexen Modellen, ermöglicht aber auch den Vergleich von verschiedenen Implementierungen derselben Funktionalität, da die verschiedenen Realisierungen durch das gemeinsame Funktionsmodell direkt miteinander verglichen werden können.

[9] Wir verwenden zur Eindeutigkeit im Weiteren den Begriff Feature für die abstrakten Funktionen des Funktionsmodells. Eine formale Definition der Features findet sich in Kapitel 5.2.1

[10] Die formalen Definitionen finden sich in Kapitel 5

- Softwaremodell
  Unser Teilmodell für die Software definiert Softwarekomponenten und Anwendungen. Anwendungen bestehen dabei aus Softwarekomponenten, diese können weiter verfeinert werden. Die Softwarekomponenten bieten entweder Schnittstellen an oder verwenden diese. Durch diese Schnittstellen werden die Abhängigkeiten zwischen den Softwarekomponenten definiert. Daraus ergeben sich die Aufrufbeziehungen für die verschiedenen Softwarekomponenten. Die Verbindung zwischen den Ebenen Software und IT-Infrastruktur ist durch die Deployment-Beziehung gegeben. Beim Deployment werden die Schnittstellen an die IT-Systeme, genauer die Endpunkte gebunden.
- Infrastrukturmodell
  Das Infrastrukturmodell beinhaltet IT-Systeme, die physikalische Komponenten der IT-Infrastruktur wie Server, Speichersubsysteme oder Netzwerkkomponenten darstellen. Diese Systeme können ineinander verschachtelt werden. Dadurch ist die Modellierung von Teilkomponenten oder virtuellen Servern möglich. Auf der Ebene der Infrastruktur existieren keine klassischen Abläufe, sondern physikalische (oder logische) Verbindungen. Diese Verbindungen zwischen den IT-Systemen werden durch Kanäle modelliert, die an Endpunkte gebunden sind. Endpunkte stellen dabei im technischen Sinne Netzwerkkarten dar. Die Kanäle sind durch eine Schnittstellenhierarchie klassifiziert, die unterschiedliche Ebenen der Kommunikation modelliert.

Die Beziehungen zwischen den Ebenen werden modelliert durch eine Abbildungen zwischen den jeweiligen Elementen aus den unterschiedlichen Teilmodellen.

Die Modelle auf den verschiedenen Ebenen und das Gesamtmodell sind minimalistisch ausgelegt und beschreiben nur die für die Modellierung der Abhängigkeiten und Abläufe notwendigen Informationen. Der Grund dafür sind bereits auf den verschiedenen Ebenen existierende Modelle (ARIS Prozessketten, Petrinetze, UML,…), die auf ihre jeweilige Ebene bezogen eine umfangreiche Modellierung erlauben. Diese sollen wie bereits beschrieben in unser Modell möglichst einfach integrierbar sein.

In der Evaluierung unseres Modells beschreiben wir ein Beispiel, das eine bereits existierende Infrastruktur analysiert und modelliert. Wir gehen dabei auf die Vorgehensweise in diesem Szenario ein.

1.6 Ausgangspunkt und Vorgehen

Wir haben bereits beschrieben, dass es mehrere Ansätze für die Modellierung von IT-Diensten gibt, je nachdem, von welcher Ebene die Modellierung ausgeht. Zu unterscheiden sind dabei Top-Down orientierte Ansätze von den Geschäftsprozessen zur IT-Infrastruktur oder Bottom-Up orientierte Vorgehensmodelle, ausgehend von den Informationen über die IT-Infrastruktur und die Verteilung der Softwarekomponenten hin zu einer Verbindung mit den Geschäftsprozessen. Relevant ist in diesem Zusammenhang aber auch der Lebenszyklus eines IT-Dienstes. Abbildung 3 verdeutlicht dies eingeordnet in unsere drei Hierarchieebenen. Zuerst werden die fachlichen Anforderungen und die Abläufe des Geschäftsprozesses beschrieben (Design), bevor dieser in Software umgesetzt (Implementierung) und anschließend auf der IT-Infrastruktur ausgeführt wird (Betrieb). Das hier angedeutete, vereinfachte Wasserfallmodell dient nur der Illustration, meist wird über den Lebenszyklus eines IT-Dienstes ein iteratives Vorgehensmodell eingesetzt werden.

Das Modell orientiert sich an diesem Ansatz. Die Aufteilung des Softwareentwurfsprozesses in die Phasen Design, Implementierung und Betrieb beschreibt die Schnittstellen zwischen den Ebenen Geschäftsprozess, Software und IT-Infrastruktur, die durch das Modell miteinander verbunden werden.

Im Gegensatz zu [Thiele05] geht diese Arbeit nicht davon aus, dass sich ein Modell der IT-Dienste einer Unternehmung im Sinne eines Bebauungsmanagements in einem reinen Top-Down Ansatz ohne eine Integration der konkreten Lösungen planen lassen. Wir gehen in einem späteren Abschnitt nochmals genauer auf die Unterschiede ein (Siehe Kapitel 2.3.5). Auch eine Verwendung eines reinen Bottom-Up Ansatzes und die Erstellung eines gemeinsamen Welt-

modells für einen IT-Dienst, das alle Informationen in ein Modell verbindet, erscheint nicht sinnvoll, da diese Modelle viel zu umfangreich, nicht mehr handhabbar und für Verantwortliche auf den Ebenen Software und Geschäftsprozess nicht mehr verständlich sind.

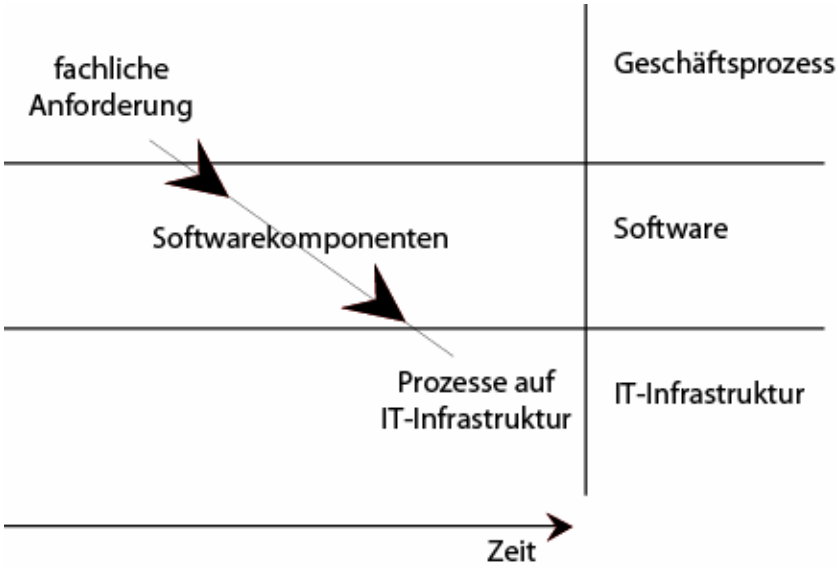

Abbildung 3 – Lebenszyklus eines IT-Dienstes und Verantwortlichkeiten

Wir fokussieren und entwickeln unser Modell auf den Softwareentwicklungsprozess bzw. den Prozess der Einführung von Anwendungssystemen auf einer IT-Infrastruktur. Diese beiden Prozesse beschreiben jeweils auch den Übergang der Ebenen. Beim Softwareentwurf werden die fachlichen Anforderungen in Software umgesetzt oder eine Kaufsoftware entsprechend adaptiert. Bei der Einführung einer Anwendung in einem Rechenzentrum wird dann diese Anwendung auf einem oder mehreren IT-Systemen installiert. Wir beschreiben am Ende des Kapitels aber auch ein umgekehrtes Vorgehen zur Erstellung eines Modells eines IT-Dienstes. Beide Vorgehensweisen sind mit unserem Modell möglich, die Wahl des Vorgehens hängt davon ab, ob der IT-Dienst bereits realisiert ist oder sich erst in der Planungsphase befindet.

Durch unser Vorgehen der Integration der Teilmodelle der verschiedenen Ebenen miteinander wird die beim Entwurf auf der jeweiligen Ebene sinnvolle Abstraktion teilweise wieder aufgehoben. Dies widerspricht auf dem ersten Blick dem Konzept der Abstraktion, das sich im Software Engineering etabliert hat. Es ist jedoch wichtig, die Ebenen miteinander zu verbinden, da die Software auf einer konkreten Infrastruktur ausgeführt werden soll und sie deren Basisdienste nutzt oder aber Basisdienste voraussetzt. Nur so können die vertikalen Abhängigkeiten sichtbar gemacht werden. Wichtig ist hier, dass wir die Abstraktion nur an den Stellen aufheben, an denen sie für dies für die jeweils anderen Ebenen notwendig sind. Implementierungsdetails, wie die Umsetzung der inneren Funktionalitäten der Softwarekomponenten, bleiben auch in unserem Ansatz für die anderen Ebenen verborgen. Dies ist vor allem auch deshalb wichtig, da durch eine vollkommene Aufhebung der Abstraktion zu umfangreiche Teilmodelle entstehen würden, deren Informationen für die Modellierung der Abhängigkeiten und Beziehungen nicht notwendig sind. Durch die vertikalen Abhängigkeiten zwischen den verschiedenen Ebenen eines IT-Dienstes können sich auch Anforderungen an die Entwicklung und das Design der Anwendung ergeben. Die Integration dieser Anforderungen in den Softwareentwurfsprozess wird durch das Modell deutlich vereinfacht. Abbildung 4 verdeutlich die verschiedenen Teilmodelle innerhalb eines iterativen Softwareentwurfsprozess eines Spiralmodells. Dies verdeutlich vor allem auch, dass Dienste meist nicht einmal entwickelt und für immer betrieben werden, wie das angedeutete vereinfachte Wasserfallmodell der Abbildung 3 suggeriert, sondern über ihre Lebenszeit verändert und weiterentwickelt werden.

Die Abbildung 4 verdeutlich den Softwarelebenszyklus in einem Spiralmodell. Am Anfang der Softwarenentwicklung oder der Evaluierung von Kaufsoftware steht die Phase des Requirement Engineerings. Hier werden die Geschäftsprozesse definiert, die durch Software unterstützt werden sollen. Darüber hinaus werden deren Anforderungen festgelegt. Wichtige Eigenschaften für den Betrieb sind hier Anforderungen wie Verfügbarkeit, Geschwindigkeit der Abarbeitung wie Transaktionsdurchsatz und Anforderungen im Bezug auf Ausfallsicherheit, die in den Service Level Agreements hinterlegt werden. Die identifizierten und beschriebenen Geschäftsprozesse werden dann auf Softwarekomponenten abgebildet, die diese ausführen. Dabei werden de Funktionen/Features[11], die die jeweiligen Softwarekomponenten erfüllen, festgelegt. Die Geschäftsprozesse werden in unserem Geschäftsprozessmodel abgelegt, die Features im Funktionsmodell. Sind die Geschäftsprozesse bereits zu einem früheren Zeitpunkt mit einer anderen Modellierungstechnik wie ARIS beschrieben worden, erfolgt hier nur eine Abbildung der betroffenen Geschäftsprozesse aus dem ARIS Modell auf die Softwarekomponenten (Siehe dazu auch 6.4). Die zu entwickelnden Softwarekomponenten werden im Softwaremodell beschrieben und die Beziehungen zu den Geschäftsprozessen und Funktionen modelliert. Damit erreichen wir eine formale Beschreibung der Zusammenhänge zwischen diesen drei Ebenen Software, Geschäftsprozess und Funktion, die für den weiteren Softwarelebenszyklus zur Verfügung stehen. Wird keine neue Software entwickelt, sondern eine bestehende Anwendung adaptiert oder eine Kaufsoftware angepasst, können durch diese Abbildung auch bereits Einschränkungen erkannt werden. So lassen sich bei Einschränkungen für einen Geschäftsprozess durch eine Kaufsoftware die Konsequenzen besser in die Entscheidung mit einbeziehen. Dadurch lässt sich die Frage beantworten, ob diese Software zum Einsatz kommen kann oder nicht, oder der Geschäftsprozess kann sehr früh entsprechend angepasst werden, um auf diese Einschränkungen zu reagieren. Eventuell mit der Kaufsoftware gelieferte Modelle über die Struktur der Anwendung können dabei in das Gesamtmodell übernommen werden.

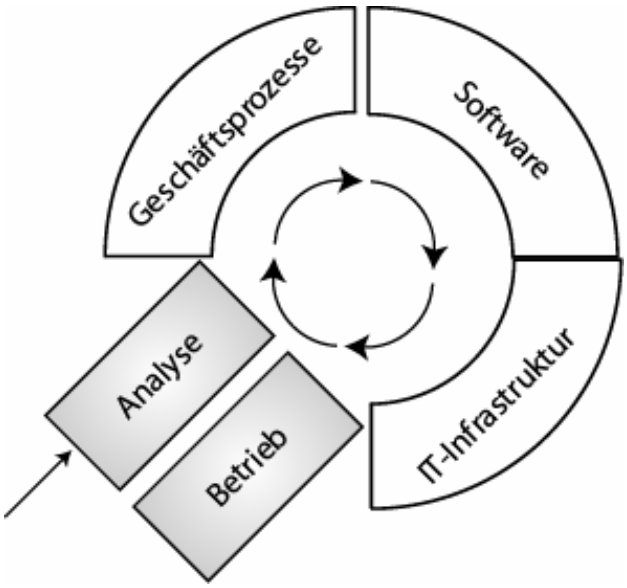

Abbildung 4 – Modellebenen im Softwarelebenszyklus

Am Übergang von der Entwicklung und Implementierung der Geschäftsprozesse in Software durch den Programmierer und der Bereitstellung und Ausführung auf der IT-Plattform durch einen Administrator findet der zweite Übergang zwischen verschiedenen Ebenen statt. Entwickelte oder gekaufte Software wird auf der IT-Infrastruktur eingerichtet und in den produktiven

[11] Wir sprechen im Weiteren von Features innerhalb de Funktionsmodells, siehe auch Kapitel 5.2.1, um Features des Funktionsmodells und Funktionen des Modells zu unterscheiden.

Betrieb übernommen. Gleichzeitig geht die Verantwortung für die Software von den Entwicklern zu den Administratoren über. In dieser Phase werden Entscheidungen der Softwarearchitekten konkretisiert und auf die IT-Infrastruktur umgesetzt. Da Software meist mit bereits vorhandenen Systemen interagieren muss, ist es wichtig, dem Entwickler die Möglichkeit zu geben, gegenseitige Abhängigkeiten von Softwarekomponenten zu beschreiben und diese Informationen weiterzugeben. Dazu gehören sowohl gegenseitige Nutzungsbeziehungen von Softwarekomponenten, aber auch Anforderungen einzelner Komponenten an die Infrastruktur. Momenten werden diese Einschränkungen und Voraussetzungen für den Administrator meist in textueller Form weitergegeben und sind dabei meist ungenügend oder für den Administrator nur schwer verständlich, da sie sich zu stark an Entwickler richten, die das System integrieren oder erweitern möchten. Hier ergeben sich neue Anforderungen an Softwareentwickler, der Techniken benötigt, um diese Informationen zu beschreiben und mit seiner Anwendung weiterzugeben. Der Begriff der Operationally-Aware Application [Daw05] von Gartner beschreibt diese zusätzlichen Anforderungen recht genau. Durch Service Descriptions müssen die Anforderungen an die Umgebung beschrieben werden, Performance Behaviour beschreibt das Verhalten der Anwendung zur Laufzeit und wie der Administrator dieses beeinflussen kann. Durch die Forderung, Anwendungen sollen Stateless sein oder ein Captured State soll durch die Anwendung realisiert werden, wird die Verlagerung einer Anwendung von einem System auf ein anderes deutlich vereinfacht, da nur die Anwendung an sich verschoben wird. Dadurch wird die Bindung zwischen Software und Hardware deutlich gelockert. Umfassende und abstrakte Möglichkeiten der Modellierung dieser Anforderungen stehen dem Programmierer dazu heute jedoch nicht zur Verfügung. Dadurch müssen sich Administratoren in die meist in unstrukturierter Form bereitgestellt Beschreibung einarbeiten und daraus die Anforderungen an ihre existierende Infrastruktur ableiten. Die Beschreibung dieser Infrastruktur wird von den Administratoren ohne Unterstützung durch den Entwickler durchgeführt. Durch die fehlende Integration der verschiedenen Modelle ergeben sich Fehlerquellen und die Aktualisierung der verschiedenen Modelle ist deutlich schwieriger. Durch die fehlende Beschreibung der Abhängigkeiten und der (oft) unbekannten Auswirkungen der Installation von neuer Software sind Administratoren bei der Einführung von neuen Anwendungen zu Recht vorsichtig, um vorhandene Systeme nicht zu beeinträchtigen. Hier setzt unser Modell an und ermöglicht eine Integration der Modelle der verschiedenen Entwicklungsstufen im Softwarelebenszyklus. Durch das vollständige Modell des IT-Dienstes hat ein Administrator so alle Informationen über den momentanen Zustand und die momentane Verteilung von Software und den entsprechenden Geschäftsprozessen auf der IT-Infrastruktur. Diese Informationen können dann als Basis für die Einführung und Verteilung neuer Komponenten sein, sind aber auch für die Behandlung von Fehlern in der IT-Infrastruktur wichtig, um deren Konsequenzen für die verschiedenen Prozesse abzuschätzen.

Ein zweiter Ausgangspunkt zur Entwicklung unseres Modells ist, wie schon beschrieben, die Erstellung des Modells für eine bereits existierende Infrastruktur. In diesem Fall wird der Softwarelebenszyklus gewissermaßen auf den Kopf gestellt. Oft sind gerade bei Legacy-Applikationen die Informationen und der Source-Code aus der Entwicklung nicht mehr vorhanden und die Erfassung der Komponenten muss durch die Analyse der Verteilung und manuelle Modellierung erfolgen. Ein Beispiel für diese Art von Vorgehen sind die ITIL Prozesse [ITIL]. ITIL beschreibt Prozesse zum IT-Management. Dabei wird ein Modell der IT-Infrastruktur durch das Release- und das Configuration-Management erfasst und gepflegt. Es setzt damit nicht bei der Entwicklung von Softwarekomponenten, sondern beim Deployment an. ITIL ist gerade in großen Unternehmen heute weit verbreitet. Es existieren unter anderem so genannte Application Scanner, die in einer vorhandenen IT-Infrastruktur Softwarekomponenten sowie deren Konfiguration erkennen können. ITIL wurde sehr stark aus der Perspektive des Administrators entwickelt. Es gibt erste Ansätze auch die Ebene der Geschäftsprozesse zu integrieren, momentan sind diese jedoch in der Entwicklung. Darüber hinaus existieren bei einigen Tools Erweiterungen, die auch eine teilweise Integration von Tests aus dem Softwareentwurfsprozess ermöglichen. Wir gehen darauf in Kapitel 2.3.1 genauer ein.

Wie bereits beschrieben kann unser Modell bei beiden Vorgehensweisen eingesetzt werden, um die Ebenen miteinander zu verbinden. Möglich ist dies, weil sich nur die Herangehensweise an die Erstellung des Modells ändert, das Modell, das die Abhängigkeiten und Beziehungen zwischen den Ebenen beschreibt, aber nicht. Unser IT-Dienstmodell kann deswegen auch dazu

verwendet werden, beide Herangehensweisen auf der Modellebene miteinander zu verbinden und so zu einem Gesamtmodell führen, das die detaillierten Teilmodelle der verschiedenen Ebenen miteinander verbindet.

1.7 Beitrag des Modells

Wir haben unser Modell und seine Struktur vorgestellt. Zusammengefasst soll unser Modell folgenden Mehrwert für das Management von IT-Diensten bieten:

- Minimale Teilmodelle zur Beschreibung der für die Modellierung der Beziehungen notwendigen Elemente der Ebenen Geschäftsprozess, Software und IT-Infrastruktur
- Modellierung der Beziehungen zwischen Elementen auf einer Ebene und der Beziehungen zwischen den Ebenen Geschäftsprozess, Softwarekomponenten und IT-Infrastruktur eines IT-Dienstes
- Bereitstellung eines Funktionsmodells zum Vergleich von verschiedenen Realisierungen eines IT-Dienstes und zur einheitlichen Beschreibung der Funktion unterschiedlicher Komponenten innerhalb einer komplexen IT-Infrastruktur
- Verknüpfung zu bereits existierenden Modellen wie ARIS zur Integration vorhandener Modelle eines IT-Dienstes und der Entwicklung einer Gesamtsicht
- Ableitung und Wiederverwendung von Wissen aus der Softwareentwicklung zur Vereinfachung der Erstellung eines integrativen Modells und zur Entwicklung eines Modells, das den gesamten Lebenszyklus eines IT-Dienstes abdeckt
- Konsistenzsicherung des Modells durch die Bereitstellung von Funktionen zur Erstellung des Modells
- Möglichkeit zur Modellierung von Ausfällen und deren Auswirkungen auf die Struktur eines IT-Dienstes
- Verwendung eines Basismodells, das die Anwendung auch ohne die Verwendung von rechnergestützten Tools ermöglicht und eine einfache grafische Darstellung beinhaltet
- Entwicklung einer gemeinsamen Basis für den Administrator, Softwareentwickler und Geschäftsprozessverantwortlichen zur Planung, Entwicklung und Pflege von IT-Diensten.

Im Gegensatz zu den Ansätzen der Softwarekartographie oder anderen rein Top-Down orientierten Ansätzen beziehen wir die IT-Infrastruktur und die technische Realisierung sehr stark ein. Nur dadurch lässt sich ein vollständiges Bild eines IT-Dienstes entwickeln, das auch Ausfälle zum Beispiel eines Servers und dessen Auswirkungen beschreiben kann. Dies ist darüber hinaus die Voraussetzung, um das Modell in ein entsprechendes Modell zum Management von IT-Diensten integrieren zu können, da hier ein Verständnis für die verschiedenen Einzelkomponenten notwendig ist.

Im Gegensatz zu den Bottom-Up Ansätzen wie ITIL integrieren wir aber auch die Perspektive der Geschäftsprozesse. Eine Verbindung unseres Modells mit den Modellen der ITIL-Tools erlaubt so eine automatische Pflege des Modells, wenn durch einen Application Scanner die Infrastruktur analysiert wird, bzw. sich die Software-Verteilung durch einen Change-Management Prozess ändert. All diese Änderungen werden dann automatisch durch die Abbildung der Modelle untereinander synchronisiert.

Durch die einfachen Teilmodelle ermöglichen wir die Integration von Modellen, die als Ergebnis eines Top-Down und eines Bottom-Up Ansatzes entstanden sind, zu einem einheitlichen Gesamtmodell.

Durch die Einbettung in den Softwareentwurfsprozess wird zum einen die Erstellung des Gesamtmodells eines IT-Dienstes vereinfacht, darüber hinaus entsteht so ein einheitliches Modell eines IT-Dienstes über dessen gesamten Lebenszyklus. Dies kann dazu beitragen, dass Wis-

sen über eine bestehende Anwendung und deren Struktur nicht verloren geht, ein Problem, dass gerade beim Software Reengineering von Legacy-Applikationen häufig auftritt.

1.8 Aufbau der Arbeit

Die Arbeit beschreibt das Modell mit seinen Teilmodellen und gibt Beispiele seiner Nutzung. Wir führen dabei zuerst in das Szenario und die Rahmenbedingungen für das Modell ein. Dabei gehen wir auf aktuelle Techniken zum IT-Management, deren Prozesse und Tools ein. Darüber hinaus stellen wir die verschiedenen Stakeholder vor, die an der Erbringung von IT-Diensten beteiligt sind und die wir schon bei der Einführung der Zielgruppe des Modells definiert haben. Daraus leiten wir die Anforderungen für die Modellierung ab.

Wir beschreiben das Modell und seine Funktionen und illustrieren dies an Bespielen. Ein komplexeres Beispiel wird zur Verdeutlichung vollständig beschrieben. Abgerundet wird die Arbeit durch die Vorstellung einer Evaluierung des Modells an einem konkreten Geschäftsprozess bei Siemens Business Services und der Auswertung dieser Evaluierung.

Ein Ausblick zeigt weitere Forschungsmöglichkeiten und Erweiterungen für das Modell.

2 Grundlagen und Szenario

Wir haben in der Einleitung bereits kurz unser Modell, seine Struktur, Teilelemente und die Vorgehensweise bei der Erstellung beschrieben. Darüber hinaus haben wir den durch unser Modell zu erzielenden Mehrwert vorgestellt.

Ausgehend von der kurzen Einführung in dieses Szenario in der Einleitung wollen wir in diesem Kapitel die Darstellung des Szenarios, in dem sich die Arbeit bewegt, vertiefen und auf verschiedene Grundlagen und Techniken für das Management von IT-Diensten eingehen. Darüber hinaus stellen wir existierende Ansätze zur Modellierung von IT-Diensten vor. Darauf aufbauend entwickeln wir in den folgenden Kapiteln dann die Anforderungen an unser Modell.

2.1 Grundlagen

Die Informationstechnologie (IT) und die mit IT realisierten Geschäftsprozesse stellen heute meist das Rückrat eines Unternehmens dar. Daraus ergibt sich auch die Bedeutung der IT-Infrastruktur für das Unternehmen. Das Management von IT-Diensten ist eingebettet in die Managementhierarchie und Managementstruktur eines Unternehmens. Die Bedeutung der IT-Infrastruktur zeigt sich heute auch daran, dass in Unternehmen die IT-Organisation und die Organisationsabteilung in einer Organisationseinheit zusammengefasst sind.

Wie bereits im Kapitel Einleitung beschrieben, lassen sich IT-Dienste in unterschiedliche Ebenen aufteilen. Die fachliche Ebene beschreibt die Geschäftsprozesse an sich in einer Form, die zur direkten Implementierung auf einem IT-System nicht geeignet ist, die aber für die beteiligten Personen den Geschäftsprozess in seiner Struktur beschreibt. Dabei werden die einzelnen Aktionen und Abläufe eines Geschäftsprozesses beschrieben. Diese müssen auf einer IT-Infrastruktur entsprechend umgesetzt werden. Dazu werden Anwendungen und Softwarekomponenten eingesetzt, die die verschiedenen Schritte der Geschäftsprozesse auf den IT-Systemen ausführen können.

2.2 Einordnung in das Management eines Unternehmens

Das Ziel unseres Modells ist die Verbesserung des Managements von IT-Diensten. Das Management der IT-Infrastruktur ist dabei ein untergeordneter Prozess des Unternehmensmanagements. Deshalb ist zuerst ein Verständnis für die verschiedenen Managementebenen wichtig, da sich aus dieser Hierarchie Anforderungen für das Management von IT-Diensten ergeben.

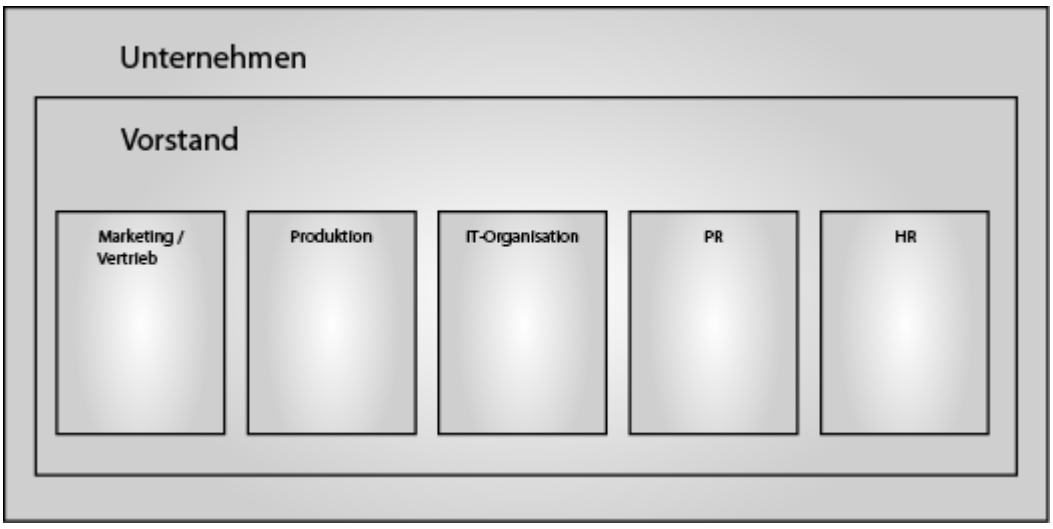

Abbildung 5 – Vereinfachte Unternehmensstruktur

Abbildung 5 zeigt eine vereinfachte Unternehmensstruktur. Die IT-Organisation realisiert dabei Dienste für die verschiedenen Fachabteilungen. In vielen Unternehmen ist die IT deswegen als interner Dienstleister konzipiert[12], entsprechend einer modularen Unternehmensstruktur.

Die bereits angedeutet Managementhierarchie wird unterstützt durch entsprechende Vorgaben und Frameworks, die auf den verschiedenen Ebenen existieren. Abbildung 6 zeigt eine beispielhafte Hierarchie von Managementframeworks, die unterschiedliche Managementebenen adressieren.

Im Rahmen dieser Einteilung realisiert das IT-Management die technische und organisatorische Verwaltung der IT-Systeme, gesteuert durch die Vorgaben und definierten Verantwortlichkeiten des IT-Governance, dessen Vorgaben sich unter anderem aus den Corporate Governance Vorgaben ergeben.

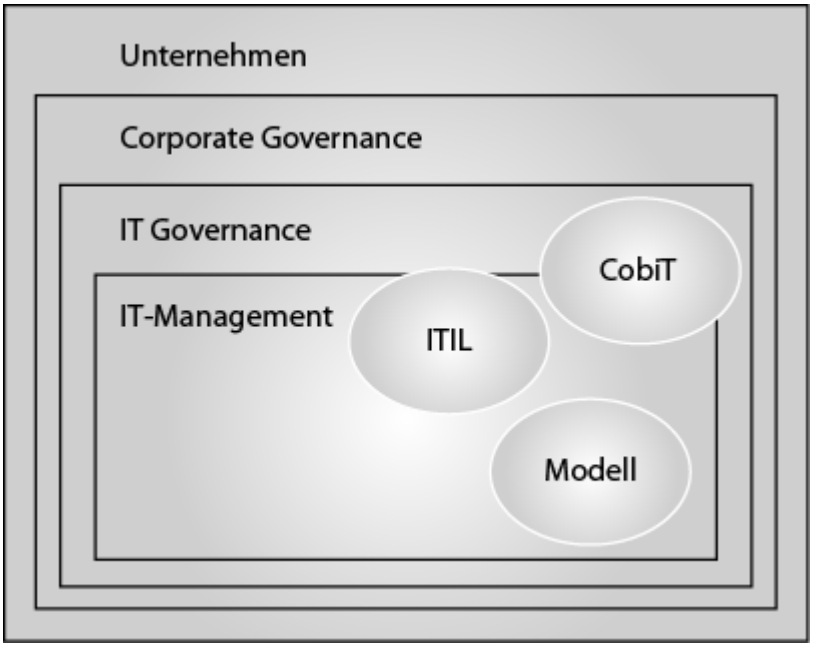

Abbildung 6 – Einordnung des Managements von IT-Diensten in Managementframeworks

Entsprechend dieser Einordnung ergeben sich von den verschiedenen Hierarchiestufen Anforderungen und Maßnahmen, die auf den untergeordneten Ebenen umgesetzt werden müssen. Ein Beispiel dafür die die Dokumentation von Geschäftsprozessen sowohl auf der fachlichen Ebene, als auch die Umsetzung in IT-Systemen im Bezug auf das Risikomanagement des Unternehmens.

2.2.1 Corporate Governance

Auf der Ebene des Unternehmensmanagements ist ein Beispiel für ein entsprechendes Framework der Deutsche Corporate Governance Kodex. Die Funktion des Kodex ist in der Präambel dargestellt:

> *„Der vorliegende Deutsche Corporate Governance Kodex (der "Kodex") stellt wesentliche gesetzliche Vorschriften zur Leitung und Überwachung deutscher börsennotierter Gesellschaften (Unternehmensführung) dar und enthält international und national anerkannte Standards guter und verantwortungsvoller Unternehmensführung.“ [Cro05].*

Definition 1 – Deutscher Corporate Governance Kodex

[12] Häufig finden sich in Unternehmen neben einer zentralen IT-Abteilung auch IT-Abteilungen in den einzelnen Fachbereichen. (Ein Beispiel dafür ist die BMW AG)

Er stellt damit eine wichtige Hilfe zur Unternehmensführung dar und integriert entsprechende gesetzliche Vorschriften und Regelungen. Diese Regelungen betreffen aber nicht nur die Managementebene an sich, sondern müssen auf allen untergeordneten Ebenen ebenfalls entsprechend umgesetzt werden.

Rechtliche Rahmenbedingungen schreiben zum Beispiel vor, wie lange Dokumentationen von Prozessen und Geschäftsvorfällen aufbewahrt werden müssen. Werden diese Prozesse in IT-Systemen umgesetzt, ergeben sich entsprechende Anforderungen an die sichere Archivierung der Daten nicht nur im Hinblick auf die Möglichkeit, die Dokumente nach (einer eventuell langen Zeit) wieder zur Verfügung zu stellen, sondern auch sicherzustellen, dass diese nicht verändert wurden. Diese Anforderungen müssen sich in entsprechenden Maßnahmen auf der Ebene des IT-Managements widerspiegeln. Gerade an diesem Beispiel zeigt sich, dass diese relativ einfache Anforderung der Archivierung eine Vielzahl an Maßnahmen auf der Ebene des IT-Managements nach sich zieht, die sowohl die Sicherheit der Dokumente als auch die Verwertbarkeit umfasst, um die Anforderung zu erfüllen.

Ein momentan vieldiskutiertes Beispiel für rechtliche Vorgaben sind die Regelungen des Sarbanes-Oxley-Act (SOX), auf den wir in einem späteren Abschnitt noch genauer eingehen. SOX wurde entwickelt, um das Vertrauen der Anleger in die Bilanzinformationen von Unternehmen nach den Pleiten von Enron und Worldcom wieder zu stärken. Die Vorgaben, die SOX dabei macht, beeinflussen vor allem auch deshalb die IT, weil sich ein Großteil der für die Erfassung und Dokumentation von Finanzinformationen eines Unternehmens eingesetzten Verfahren auf IT-Systemen abstützt. Deshalb gelten Vorgaben für die entsprechenden Verfahren auf der fachlichen Ebene auch für die technische Ebene, durch die diese realisiert werden.

2.2.2 IT-Governance

IT-Governance stellt ein Rahmenwerk für die Organisation der IT-Abteilung dar, das die Regeln, Rollen und Verantwortlichkeiten für die IT-Abteilung festlegt. [Krcmar] definiert IT-Governance nach Weill/Woodham:

> *„Specifying the decision rights and accountability framework to encourage desirable behaviour in the use of IT“ [Weill02]”*

Definition 2 – IT-Governance

IT-Governance stellt somit die Basis und Spielregeln für das konkrete, technische IT-Management und die daran beteiligten Personen dar. Es regelt deren Rechte und Pflichten und gibt so einen Handlungsrahmen für die verschiedenen Personen und Rollen vor.

Gerade rechtliche Rahmenbedingungen beeinflussen, wie schon aufgezeigt, in zunehmendem Maße Entscheidungen auf der Ebene der IT-Systeme und stellen Anforderungen sowohl an die Organisation, Dokumentation als auch an die Art und Qualität der Diensterbringung. Diese rechtlichen Rahmenbedingungen schlagen sich dann auch entsprechend in den Vorgaben des IT-Governance nieder.

Mit dem Control Objectives for Information and Related Technology (CobiT) Framework, das durch das IT-Governance Institute entwickelt und gepflegt wird, steht dafür ein Rahmenwerk zur Verfügung. In der Version 4 werden dabei 34 Prozesse definiert, die in die Bereiche Planung, Entwicklung, Betrieb und Monitoring unterteilt sind. Diese beschreiben die jeweiligen Aufgaben für die IT-Abteilung. CobiT enthält außerdem ein Reifegradmodell, mit dem sich die Umsetzung dieser Prozesse in der eigenen IT-Abteilung überprüfen lässt. Der Nutzen, den CobiT für IT-Governance durch die Veränderung der IT-Prozesse erreichen möchte ist:

- Verbesserte Ausrichtung, die auf den Unternehmenserfordernissen basiert
- Eine für das Management verständliche Sicht auf die Aktivitäten der IT
- Klare, auf Prozessen basierende Zuweisung von Ownership und Verantwortlichkeit
- Allgemeine Akzeptanz bei Drittparteien und Regulatoren

- Gemeinsames Verständnis bei allen Stakeholdern, das auf einer gemeinsamen Sprache basiert
- Erfüllung der Anforderung von COSO für ein Kontrollsystem über die IT

  Quelle: [CobiT4]

Gerade die Verbindung zwischen CobiT und COSO schlägt die Brücke zwischen IT-Governance und SOX. Die Regelungen von SOX stützen sich häufig auf den COSO Vorgaben ab, Vorgaben aus SOX für das IT-Governance finden sich so in CobiT wieder.

2.2.3 IT-Management in der Unternehmenshierarchie

Das IT-Management setzt durch den Einsatz von Managementwerkzeugen die konkreten Geschäftsanforderungen nach den Regeln des IT-Governance auf der IT-Infrastruktur um. Das IT-Management beschreibt dabei die Prozesse, durch die die Vorgaben technisch umgesetzt werden. Hegering [Heg99] definiert das IT-Management als:

> *„Alle Maßnahmen für einen unternehmenszielorientierten effektiven und effizienten Betrieb eines verteilten Systems mit seinen Ressourcen."*

Definition 3 – IT-Management

Es existieren verschiedene Sammlungen, die diese Regeln und Vorgaben für das IT-Management zusammenfassen, eine der wichtigsten und weit verbreiteten ist ITIL [ITIL]. ITIL spezifizieren die technischen Prozesse, die für das IT-Management innerhalb der IT-Abteilung zu definieren sind und stellt Best-practices vor, wie diese umgesetzt werden sollen. Es existieren verschiedene Varianten von ITIL, eine davon ist das Microsoft Operations Framework (MOF) [MOF], das die allgemeinen ITIL Prozesse und Vorgaben konkretisiert und auf die speziellen Aufgaben und Werkzeuge einer Microsoft basierten IT-Infrastruktur abbildet.

Der Einsatz von IT-Managementvorgaben wie ITIL trägt zur Standardisierung der IT-Prozesse bei. Wichtig hierbei ist jedoch, dass ITIL nicht eins zu eins in einem Unternehmen umgesetzt werden kann, sondern, wie schon in der Einleitung beschrieben, an die Unternehmensstruktur angepasst werden muss.

Wir werden uns in diesem Kapitel wegen der weiten Verbreitung und der guten Unterstützung durch Managementtools auf ITIL als ein Framework zum IT-Management abstützen.

2.2.4 Organisation des IT-Managements

Die Organisation der IT-Abteilung und des IT-Managements erfolgt heute in unterschiedlichen Arten. Die Aufteilung in die unterschiedlichen Ebenen variiert dabei von Unternehmen zu Unternehmen. Teilweise werden IT-Systeme als Silo gesehen und auch als solches verwaltet. Abbildung 7 verdeutlicht dies auf der rechten Seite. Hier erfolgt ein vertikales Management der IT-Dienste. Innerhalb des gedachten Silos ist hier auch eine Verknüpfung der verschiedenen Ebenen veranschaulicht. Die Verantwortlichen sind bei dieser Art des Managements für das gesamte System mit seiner Hardware und seiner Software verantwortlich. Dies ermöglicht ein ganzheitliches Management eines IT-Dienstes, erfordert aber auch, dass jeder Silo-Verantwortliche den gesamten Stack beherrscht. Gerade bei komplexen IT-Infrastrukturen ist dieser Ansatz meist nicht mehr umsetzbar.

Eine andere Form der Organisation ist die Aufteilung der Verantwortung nach Ebenen. Ein horizontales Management, wie in Abbildung 7 für die Serversysteme eingezeichnet, ermöglicht es, Wissen für eine bestimmte Ebene in einer Abteilung zu konzentrieren. So können Economy of Scale-Effekte realisiert werden, darüber hinaus ist eine bessere Kommunikation zwischen den Verantwortlichen auf dieser Ebene gewährleistet, als wenn diese auf die verschiedenen „Silos" aufgeteilt sind. Ebenso besser unterstützt wird in diesem Fall die Entwicklung und Umsetzung eines Standardvorgehens auf den einzelnen Ebenen. Damit wird sichergestellt, dass zum Beispiel alle Server mit einem speziellen Betriebsystem in gleicher Weise verwaltet werden. In diesem Szenario sind für die Verwaltung eines einzigen Dienstes jedoch mehrere Personen aus unterschiedlichen Abteilungen verantwortlich. Darüber hinaus fehlt hier das Wissen über die

Funktionsweise der anderen Schichten eines IT-Dienstes. Der Kommunikationsaufwand zwischen den verschiedenen beteiligten Personen muss in diesem Fall mit betrachtet werden.

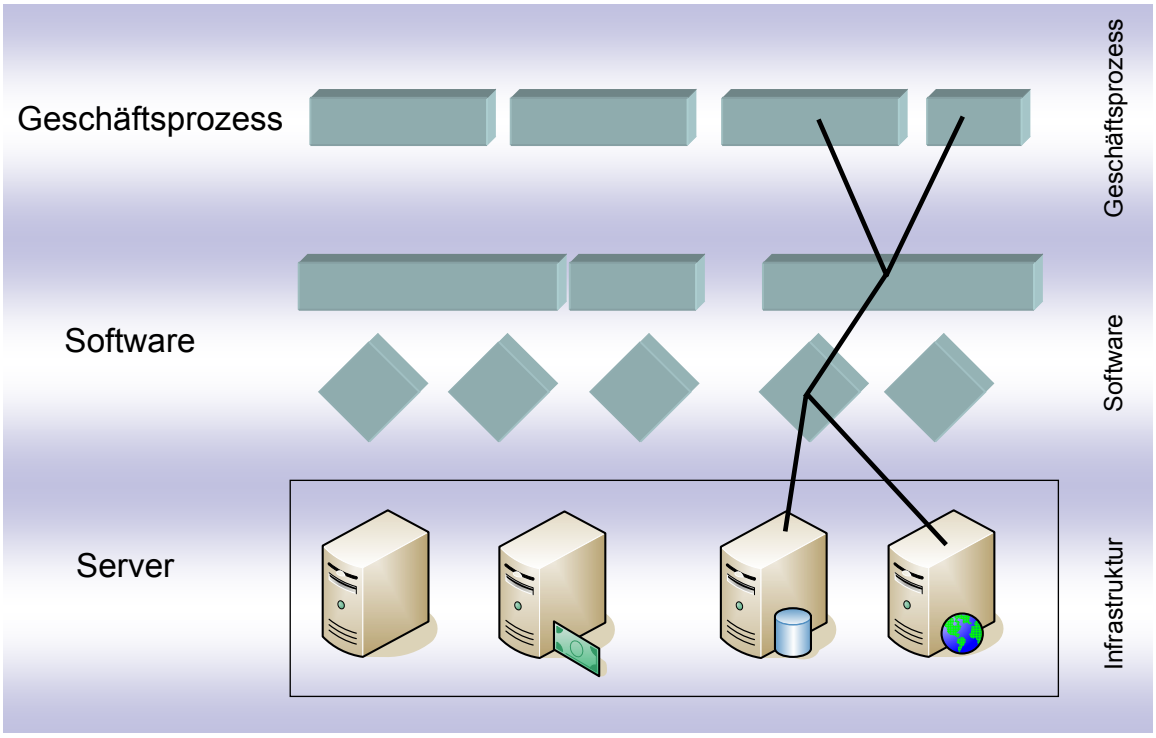

Abbildung 7 – Horizontales und vertikales IT-Management

Beide Ansätze bringen Vor- und Nachteile. Je nach Unternehmen finden sich verschiedene Ausprägungen, wobei gerade in großen Konzernen wie Siemens oder BMW die Organisation nach Ebenen häufiger zu finden ist, gerade um die Bereitstellung auf der Infrastrukturebene zu vereinheitlichen und dieser Ansatz weniger Mitarbeiter auf der IT-Ebene erfordert, da ein Mitarbeiter die IT-Systeme mehrerer Fachanwendungen betreut.

2.2.5 Vorgaben auf Unternehmensebene

Neben den Vorgaben für das Management existieren in einem Unternehmen noch andere Vorgaben, die umgesetzt werden müssen und der Steuerung oder Vereinheitlichung dienen. Dazu zählen Vorgaben, die für die Entwicklung und den Betrieb von IT-Diensten gelten und die speziell für das jeweilige Unternehmen entwickelt wurden. Ein Beispiel dafür sind Architekturvorgaben zur Entwicklung von Softwaresystemen wie der Blueprint der BMW AG. Im BMW Blueprint wird festgelegt, wie verschiedene Arten von Softwaresystemen, die bei BMW eingesetzt werden sollen, entwickelt werden müssen. Diese Vorgaben beschreiben dabei Prozesse, aber auch Vorgaben an die Architektur und an die Verteilung der Komponenten. Durch die Vorgaben soll eine Standardisierung der Softwarelandschaft, vor allem bei Eigenentwicklungen erreicht werden. Die Vorgaben berücksichtigen auch die IT-Infrastruktur und versuchen zu einer Standardisierung beizutragen, indem die Architekturpattern eine konkrete Verteilung vorgeben und die verschiedenen Versionen der Basissoftwarekomponenten, die bereits freigegebenen sind, definiert werden. Der Blueprint selbst stützt sich auf Standards wie Softwarearchitektur Pattern [GOF95] ab.

Diese Pattern sind Vorgaben an die Softwareentwicklung. Auch hier wird durch Standardisierung versucht, die Entwicklung von Softwaresystemen zu vereinfachen, Fehlerquellen zu vermeiden und möglichst einfach erweiterbare Anwendungssysteme zu entwickeln.

2.3 IT-Management Ansätze und Tools zur Realisierung

Wir haben die Aufgaben des IT-Managements bereits kurz dargestellt. In diesem Abschnitt werden wir auf ITIL genauer eingehen und die einzelnen Prozesse kurz vorstellen. ITIL beschreibt dabei vor allem das technische Management von IT-Diensten, es repräsentiert in unserer Einteilung der Vorgehensweisen zum IT-Dienstmanagement einen Bottom-up Ansatz, da es das Management von IT-Diensten aus der Perspektive der IT-Infrastruktur betrachtet.

Wir gehen am Ende des Abschnitts auch auf die Arbeit von [Thiele05] ein und beschreiben, wieso auch wir von einem eher Bottom-up orientierten Managementansatz ausgehen und einen reinen Top-Down Ansatz für nicht zielführend halten.

2.3.1 IT-Management nach ITIL – Bottom up

Das IT-Management (oder IT-Service Management) setzt die Unternehmensvorgaben und die Geschäftsprozesse auf der IT-Infrastruktur um.

ITIL [ITIL] ist ein Standard, der durch das britische Office of Government Commerce (OGC) in Norwich entwickelt und gepflegt wird. ITIL definiert Prozesse und beschreibt Best-Practices für das IT-Management und trägt damit zu dessen Standardisierung bei.

> *„ITIL ist die Abkürzung für den durch die CCTA (heute OGC) in Norwich (England) im Auftrage der britischen Regierung entwickelte Leitfaden IT Infrastructure Library. ITIL ist heute der weltweite De-facto-Standard im Bereich Service Management und beinhaltet eine umfassende und öffentlich verfügbare fachliche Dokumentation zur Planung, Erbringung und Unterstützung von IT-Serviceleistungen. ITIL bietet die Grundlage zur Verbesserung von Einsatz und Wirkung einer operationell eingesetzten IT-Infrastruktur. An der Entwicklung von ITIL waren IT-Dienstleister, Mitarbeiter aus Rechenzentren, Lieferanten, Beratungsspezialisten und Ausbilder beteiligt."*
>
> *Quelle: [ITIL-Web].*

Definition 4 – ITIL

Durch die Einführung von ITIL werden die IT-Prozesse im Unternehmen verbessert und auf einen allgemein anerkannten Standard vereinheitlicht. Mit BS 15.000 wurde ein englischer Standard entwickelt, der die Anforderungen an das IT Servicemanagement definiert.

Mit der ISO Norm 20.000 [ISO2k] wurde Ende 2005 als Weiterentwicklung des BS 15.000 ein internationaler Standard unter dem Dach der ISO/IEC für das IT-Service Management geschaffen. Die Umsetzung dieser Norm kann im Gegensatz zu ITIL im Unternehmen zertifiziert werden. Damit ist eine Zertifizierung der IT-Prozesse möglich, ähnlich einer ISO 9000 Zertifizierung. Diese Möglichkeit der Zertifizierung wird voraussichtlich noch mehr Unternehmen dazu bewegen, ITIL einzuführen.

ITIL unterscheidet drei verschiedene Ebenen des IT-Managements:

- Strategisches IT-Management, wie Qualitätsmanagement und IT-Service-Organisation
- Taktisches IT-Management, für die konkrete Planung und Steuerung von IT-Diensten
- Operatives IT-Management, für die Unterstützung und den effizienten Betrieb von IT-Diensten

Innerhalb dieser verschiedenen Ebenen definiert ITIL die konkreten Prozesse und Vorgaben für die unterschiedlichen Aufgabenbereiche des IT-Managements.

Wichtig zu beachten ist, dass auch ITIL von Prozessen spricht. Im Gegensatz zu unserem Verständnis sind hier jedoch IT-Management Prozesse und keine Geschäftsprozesse gemeint. ITIL selbst beinhaltet keine Definition von Geschäftsprozessen, diese können nur indirekt in den Informationen des Configuration Management abgelegt werden. Momentan findet eine Überarbeitung von ITIL statt. Die nächste Version von ITIL wird auch eine Sicht für das Management beinhalten. Damit soll die Brücke zwischen dem technischen IT-Management und dem Business-Prozess Engineeering geschlagen werden.

Wir stellen in diesem Abschnitt die für das Management von IT-Diensten hauptsächlich relevanten ITIL-Prozesse vor, die wir durch unser Modell besser unterstützen möchten. Wir geben einen kurzen Einblick, dies soll nur die Aufgaben des IT-Managements verdeutlichen und ist keine Einführung in ITIL. Wir betrachten dabei vor allem taktische und operative Prozesse, obwohl unser Modell auch beim Financial Management und für das Capacity Planning, wie bereits angedeutet, einen Beitrag liefern kann.

Wir gehen auch auf Funktionen wie den Service Desk, der den Helpdesk in einem Unternehmen beschreibt und organisiert nicht detailliert ein. Diese greifen über die ITIL Prozesse Incident- und Change Management auf Funktionen zum Management von IT-Diensten (im speziellen auf die Configuration Management Database) zu.

Configuration Management

ITIL definiert das Configuration Management als:

> *„The process of identifying and defining the Configuration Items in a system, recording and reporting the status of Configuration Items and Requests for Change, and verifying the completeness and correctness for configuration items."*

Definition 5 – Configuration Management nach ITIL

Das Configuration Management erfasst alle Informationen über die IT-Infrastruktur und die Softwarekomponenten und fasst die Informationen aus verschiedenen Quellen zu einem gemeinsamen Modell zusammen, das die Ebenen IT-Infrastruktur und Software beschreibt. Die Informationen über die einzelnen Bestandteile der Infrastruktur werden dabei in den Configuration Items abgelegt. Zu diesen Informationen gehören auf der Ebene der Infrastruktur die Informationen über die Eigenschaften von physikalischen Komponenten eines IT-Systems, aber auch Informationen über die Softwareverteilung auf diesem System. Darüber hinaus wird hier auch die Struktur und die Konfiguration der Netzwerkinfrastruktur gespeichert. Auch Informationen für die Inventarisierung und die Softwarelizenzierung werden hier zusammengefasst und gespeichert. Es entsteht so eine Datenbasis, die alle relevanten Informationen zur IT-Infrastruktur und Softwareausstattung beinhaltet und über die Zeit archiviert. Somit sind auch zeitliche Veränderungen nachvollziehbar.

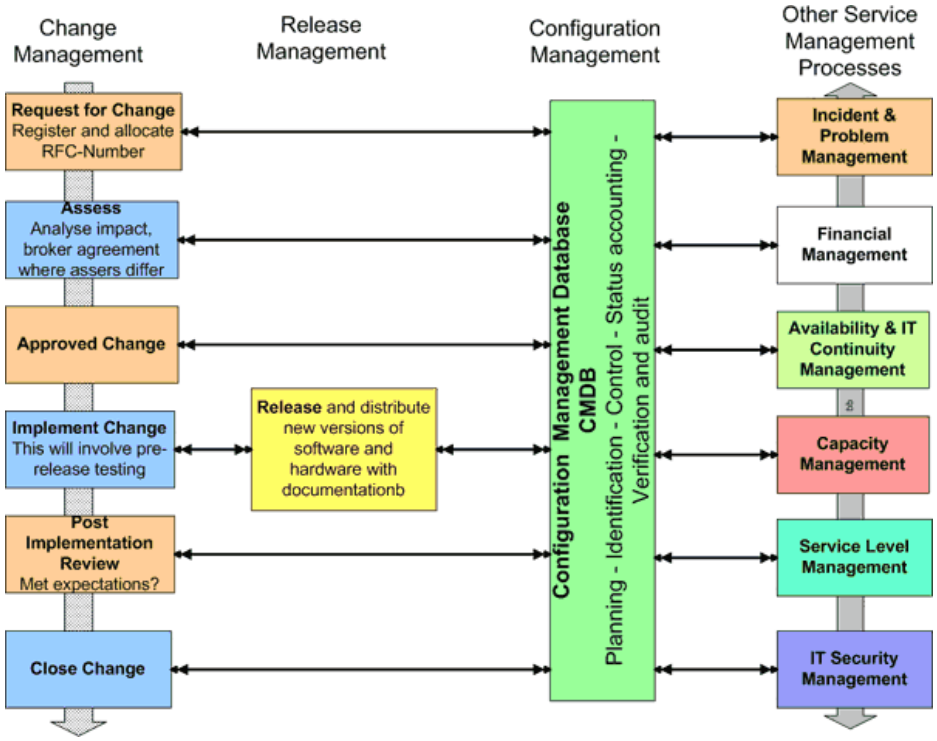

Abbildung 8 – Configuration Management nach ITIL (Quelle ITIL)

Das Configuration Management liefert folgende Funktionen für das IT-Management:

- Identifikation und Ausgabe von Informationen über die Komponenten der IT-Infrastruktur und der Anwendungen, die auf der IT-Infrastruktur ausgeführt werden. Der Zugriff auf diese Informationen wird geschützt, damit nur in kontrollierter und sicherer Form auf diese Informationen zugegriffen werden kann.
- Erfassung, Verwaltung und Aufzeichnung des Status der Komponenten der IT-Infrastruktur. Dazu werden alle Komponenten der Infrastruktur erfasst und die Informationen gespeichert. Durch die Bereitstellung von Berichten wird die Kontrolle der Informationen ermöglicht und ein Überblick über den Zustand der Komponenten gegeben

Die Datenbasis wird in einer Configuration Management Database (CMDB), gespeichert. Die CMDB realisiert ein Data Warehouse, das alle Informationen sammelt und zur Verfügung stellt. Mit eine Aufgabe des Configuration Management ist dabei auch die Definition der Inhalte der Configuration Items (CI) in der CMDB.

Der Configuration Management Prozess selbst ist sehr stark mit den anderen ITIL-Prozessen verwoben, da er für diese die Informationen ablegt bzw. bereitstellt. Die Abbildung 8 verdeutlicht den Prozess und seine Verbindungen. Vor allem das Incident- und das Change Management arbeiten auf der Datenbasis des Configuration Management und sind auf die Informationen in den CI's angewiesen.

Change Management

Der Change Management Prozess ist definiert als:

> *„The Process of controlling Changes to the infrastructure or any aspect of services, in a controlled manner, enabling approved Changes with minimum disruption."*

Definition 6 – Change Management nach ITIL

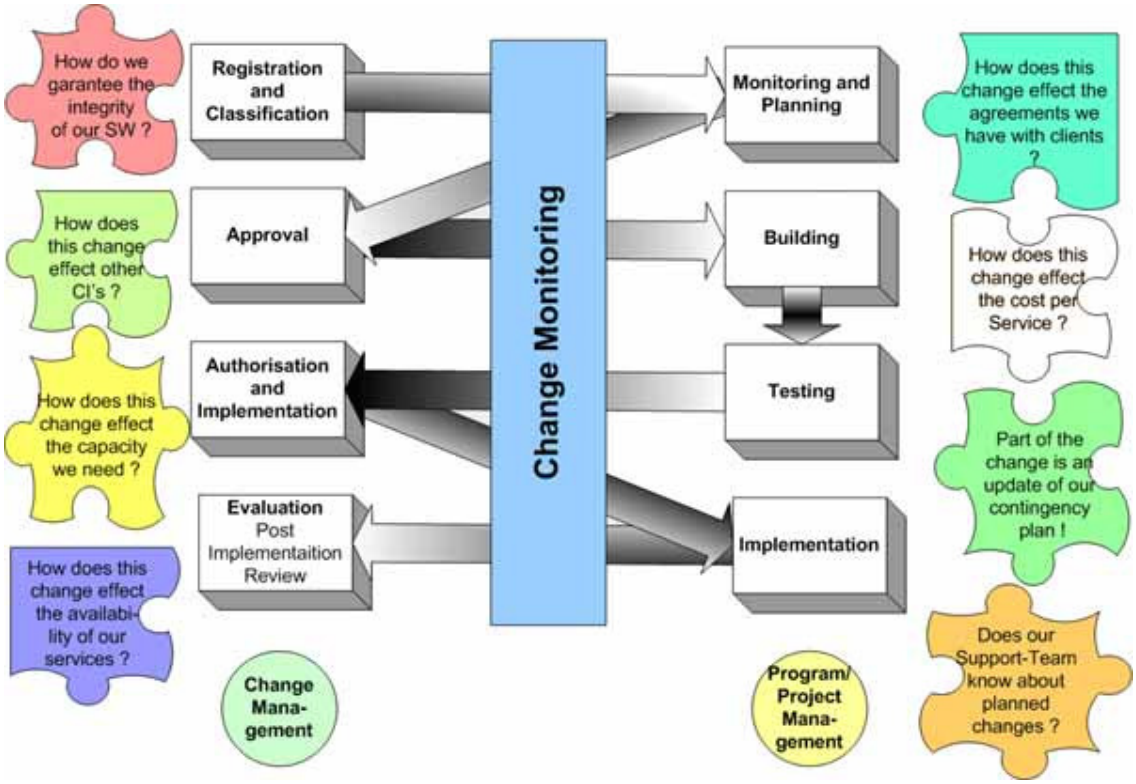

Abbildung 9 – Change Monitoring (Quelle ITIL)

Die Aufgabe des Change Management Prozesses in ITIL ist also die Verwaltung der Veränderungen an der IT-Infrastruktur und an den Softwarekomponenten. Er ist somit der wichtigste Pflegeprozess für die CMDB. Wichtig ist dabei die Definition „approved“, es handelt sich hier um einen kontrollierten Prozess, der Kontrollen und Freigaben vorsieht, im Gegensatz zum Incident Management, das auf Vorfälle wie Ausfälle direkt reagieren muss.

Zur Initiierung eines Change Request werden Request for Changes (RFC) gestellt, die die Veränderung beschreiben. Ein RFC wird im Change Management Prozess bearbeitet. Nach der Planung, welche Veränderungen für den Prozess notwendig sind und welche Veränderungen der Prozess bewirkt, müssen diese Veränderungen genehmigt werden. Anschließen werden die Veränderungen in der Software- und IT-Infrastruktur einer Testumgebung, die möglichst genau der Konfiguration der Produktionsumgebung entspricht, durchgeführt und die Ergebnisse verifiziert. Diese Ergebnisse werden dabei auch mit den erwarteten Auswirkungen abgeglichen, um mögliche Probleme zu erkennen. Nach den Tests in der Testumgebung, deren Verifizierung und der Überprüfung, ob alle gewünschten Veränderungen eingetreten sind, kann der Change Request freigegeben und auf der produktiven Infrastruktur implementiert werden. Danach müssen die Veränderungen auch auf der produktiven Infrastruktur überprüft werden, um festzustellen ob die gewünschten Veränderungen erfolgt sind. Ist dies der Fall, ist der Change Request abgeschlossen.

Der Prozess arbeitet mit und auf den Daten in der CMDB, durchgeführte Änderungen an der Software oder der Infrastruktur müssen in der CMDB gepflegt werden, um den Prozess abzuschließen. Dies stellt sicher, dass die Veränderungen an der IT-Infrastruktur oder den Softwarekomponenten auch so durchgeführt wurden, wie sie geplant waren und sorgt dafür, dass die Informationen in der CMDB auch die aktuellen Konfigurationen der Komponenten der IT-Infrastruktur und der Software darstellen.

Die Abbildung 9 verdeutlicht den Ablauf des Change Monitoring. Die Rahmenbedingungen, die an den beiden Rändern vorgestellt werden, verdeutlichen die verschiedenen Bereiche, die beachtet werden müssen.

Release Management

Der Release Management Prozess ist verantwortlich für die Einführung und die Erneuerung von Hard- und Softwarekomponenten in der IT-Infrastruktur. Er ist definiert als:

> *„A collection of new and/or changed CIs which are tested and introduced into the live environment together.“*

Definition 7 – Release Management nach ITIL

Das Release Management ist für die Erstellung, Erprobung und die Installation von neuen Applikationen verantwortlich. Darüber hinaus gehört dazu auch die Einführung und Pflege der physikalischen IT-Infrastruktur. Diese neuen CI’s werden anschließend in die CMDB eingefügt und die Konsistenz der Informationen in der CMDB mit der tatsächlichen IT-Infrastruktur, der Softwareverteilung und deren Konfiguration so sichergestellt.

Im Vergleich zum Change Management ist die Aufgabe des Release Managements die Einführung neuer Applikationen und nicht nur die Veränderung der Konfiguration, wobei beide Prozesse sehr eng miteinander in Verbindung stehen und sich teilweise überlappen können. Die Abbildung 10 verdeutlicht die einzelnen Schritte, die dabei notwendig sind. Sie stellt auch die Anforderungen an die Infrastruktur dar. Neben einer Entwicklungsumgebung, die zur Entwicklung und zum Debugging der Software verwendet wird, steht eine Test- oder Integrationsumgebung zur Verfügung, in der die Dienste unter den gleichen Bedingungen wie im produktivem Umfeld im Live Environment ausgeführt werden. Wie schon im Change Management Prozess beschrieben, können so die Auswirkungen von Veränderungen auf die ausgeführten Dienste einfach überprüft werden.

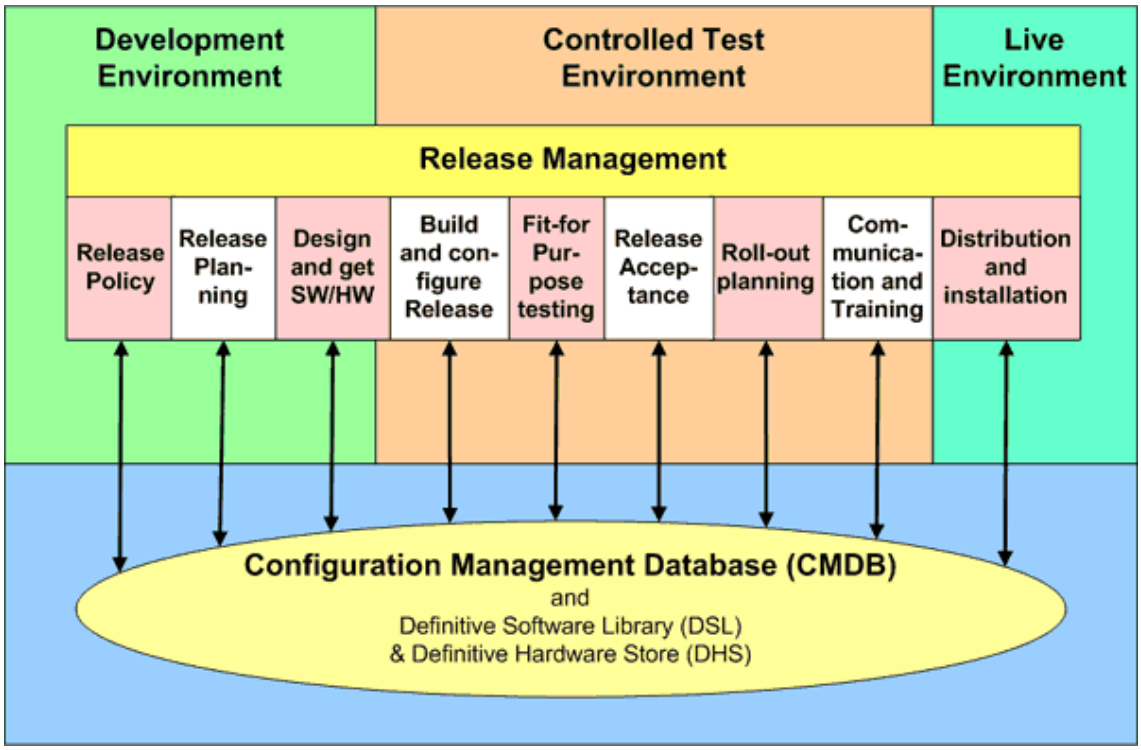

Abbildung 10 – Release Management (Quelle ITIL)

Incident Management

Ein weiterer wichtiger Prozess im täglichen Betrieb von IT-Diensten ist das Incident Management. Der Incident Management Prozess ist definiert als:

> *„Any event which is not part of the standard operation of a service and which causes, or may cause, an interruption to, or a reduction in, the quality of that service."*

Definition 8 – Incident Management nach ITIL

Das Incident Management ist damit das Gegenstück zum Change Management. Während beim Change Management Änderungen an der Infrastruktur durch eine Anfrage und einen kontrollierten Prozess mit RFC durchgeführt werden, ist der Auslöser für das Incident Management ein Ereignis, das in der Infrastruktur auftritt.

Je nach Schwere des Ereignisses, können die Reaktionen einen kontrollierten Prozess (Festplattenplatz geht zur Neige, eine Konfigurationsänderung in der Festplattengröße ist notwendig) und damit wieder einen Change-Management Prozess oder aber eine sofortige Reaktion (Kernel-Crash auf einem Server) erzwingen. Die Prozesse unterscheiden sich also vor allem im auslösenden Ereignis.

Wichtig ist hierbei vor allem der Einsatz geeigneter Überwachungs- oder Monitoringwerkzeuge, um diese Ereignisse zu erfassen und möglichst rasch auf diese reagieren zu können. Man bezeichnet dies auch als pro-aktives Service-Monitoring. Dabei wird nicht nur die Verfügbarkeit des Services selbst, sondern alle Ereignisse der Infrastruktur überwacht. Danach wird analysiert, ob sich aus den Ereignissen schwerwiegendere Konsequenzen für einzelne IT-Dienste ergeben können. Dadurch kann die Serviceverfügbarkeit gesteigert werden, weil der Ausfall des Services durch die Behebung vermieden werden kann. Für diese Analyse ist ein weit reichendes Wissen über die Beziehungen der Systeme und deren mögliche Abläufe notwendig. Heutige System integrieren dieses Wissen meist nur für wenige Applikationen des jeweiligen Herstellers. Interessant ist in diesem Zusammenhang beim Ausfall eines Services auch die Möglichkeit zur so genannten Root-Cause Analysis. Dabei wird überprüft, welches System oder welcher Dienst der Auslöser für den Ausfall eines Services war.

Availability Management

Das Availability Management ist definiert als:

> *"Availability management ensures the availability of IT services as specified by the customer"*

Definition 9 – Availability Management nach ITIL

Das Availability Management ist für die Verfügbarkeit der erbrachten Dienste zuständig. Dafür ist es wichtig, die verschiedenen Abhängigkeiten der Dienste untereinander zu kennen. Die Abbildung 11 verdeutlicht dies, auch durch die Darstellung der verschiedenen Stakeholder, die an einem entsprechenden IT-Dienst beteiligt sein können.

Eine weitere wichtige Aufgabe ist die Überwachung der Verfügbarkeit der Dienste und die Erstellung entsprechender Reports und Statistiken, die als Basis für die Abrechnung der Dienste oder die Planung und Anpassung der SLA's dienen. Diese Funktion wird auch häufig als End-to-End Monitoring bezeichnet. Dabei wird nicht nur der IT-Dienst an sich aus der Sicht des Rechenzentrums im Servernetzwerk überwacht, sondern auch sichergestellt, dass der Anwender Zugriff auf diesen Dienst hat und der Dienst für den Kunden verwendbar ist. Ein Beispiel ist eine Website. Es wird nicht nur sichergestellt, dass der Webserver ausgeführt wird und die Inhalte ausliefert, sondern es wird darüber hinaus beim Kunden ein Server zum Monitoring aufgestellt, der prüft, ob aus dem Kundennetz ein Zugriff auf den Server möglich ist und die Webseiten des Webservers für die Benutzer abrufbar sind.

Abbildung 11 verdeutlicht darüber hinaus auf die verschiedenen Regelwerke, in denen die Beziehungen rechtlich geregelt sind. Zwischen dem Benutzer und dem IT Service Provider sind dies die Service Level Agreements, die die Eigenschaften eines Dienstes definieren. Für den Service Provider intern sind es Operational Level Agreements, die Beziehungen zu Herstellern oder externen Dienstleistern beschreiben.

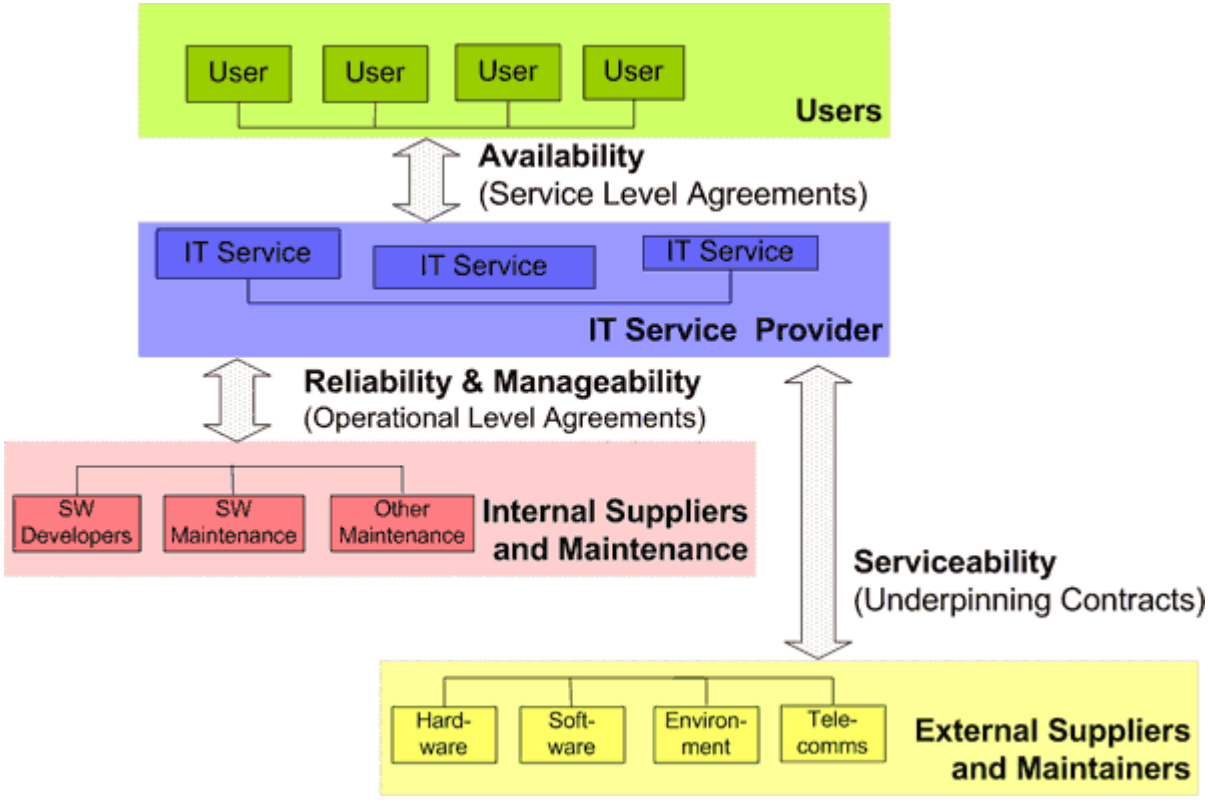

Abbildung 11 – Availability Management (Quelle ITIL)

Zusammenfassung

Wir haben verschiedene ITIL Prozesse vorgestellt. Es wird deutlich, dass alle Prozesse auf Wissen über die verschiedenen Dienste angewiesen sind. In ITIL werden diese Informationen durch das Configuration Management zur Verfügung gestellt und in einer CMDB gepflegt. Wir gehen in einem der nächsten Abschnitte auf ein Beispiel einer CMDB ein. Wir haben auf die Darstellung von ITIL Prozessen zum Financial Management oder zum Capacity Management verzichtet, auch diese können durch unser Modell unterstützt werden.

ITIL macht keine Vorgaben über die konkrete Ausgestaltung von Tools zum IT-Management. Es beschreibt auch kein vollständiges Modell zur Beschreibung von IT-Diensten inklusive der Geschäftsprozesse, alle diese Informationen würden nach ITIL in den CI's der CMDB abgelegt und durch die ITIL Prozesse gepflegt.

Eine zentrale Idee unseres Modells ist die Verknüpfung zwischen Softwareentwicklung und Softwarebetrieb. Wir möchten so zu einem umfangreichen Softwarelifecyclemanagement beitragen. ITIL benutzt im Gegensatz dazu ein IT-Management-zentriertes Vorgehen und erfasst die Softwarekomponenten durch das Release Management bzw. durch das Configuration Management bei der Einführung auf der IT-Infrastruktur. Dieses Release-Management ist dabei nicht direkt mit der Softwareentwicklung verbunden[13].

Wir werden im nächsten Abschnitt die Vorgehensweise von ITIL in einem abstrakteren Prozess zur Erstellung eines IT-Dienstmodells zusammenfassen.

2.3.2 Bottom-up Prozess zur Erstellung eines IT-Dienstmodells

Wir stellen in diesem Abschnitt ein Beispiel für einen allgemeinen Prozess zur Erstellung eines Modells eines IT-Dienstes vor, der auch das ITIL Vorgehen abdeckt. In Kapitel 8 beschreiben wir diesen Prozess für unser Modell.

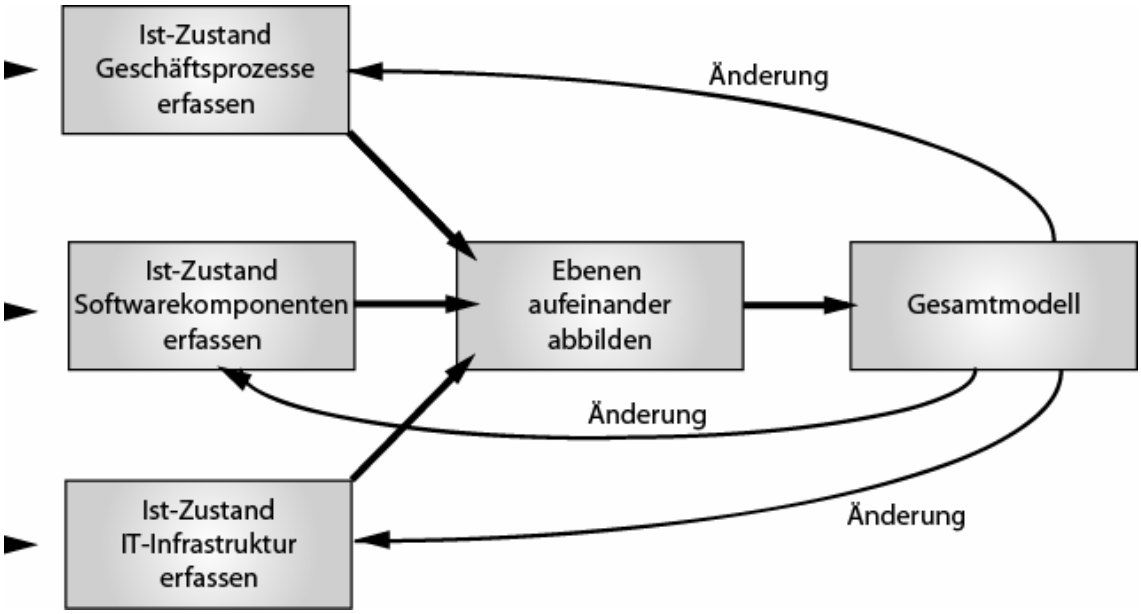

Abbildung 12 – Erstellung eines IT-Dienstmodells in einer existierenden IT-Infrastruktur

Abbildung 12 verdeutlich diesen allgemeinen Prozess. Zuerst muss der Ist-Zustand der verschiedenen Ebenen analysiert und erfasst werden. Dabei werden die vorhandenen Elemente auf der jeweiligen Ebene erfasst, ihre Abhängigkeiten analysiert und diese in einem Teilmodell beschrieben. Anschließend werden die horizontalen Abhängigkeiten erfasst und die Elemente

[13] Einige Werkzeuge bieten hier Erweiterungen

der unterschiedlichen Ebenen miteinander verknüpft. So werden die Beziehungen zwischen den Teilebenen modelliert. Daraus entsteht das Gesamtmodell, das den IT-Dienst vollständig beschreibt.

Wir haben bereits im Abschnitt über ITIL gesehen, dass ein entsprechendes Modell nicht statisch sein kann. Es ändert sich über die Zeit durch die Einführung neuer Komponenten oder Veränderungen an der Konfiguration existierender Komponenten. Entsprechend muss das Gesamtmodell iterativ gepflegt werden, um die Realität der IT-Infrastruktur entsprechend abzubilden.

Wir gehen im nächsten Abschnitt auf Tools zum Management von IT-Diensten auf den verschiedenen Ebenen ein. Unter anderem stellen wir hier die Tools zum Management nach ITIL vor.

2.3.3 Managementtools

Wir stellen in diesem Abschnitt aktuelle Managementtools vor und beschreiben so heute verfügbaren Möglichkeiten zum Management von IT-Diensten. Wir beschreiben dabei sowohl Tools, die sich auf spezielle Ebenen konzentrieren, als auch Werkzeuge, die spezielle ITIL Prozesse unterstützten.

Management von IT-Systemen

Das Management von Hardware auf der Ebene von IT-Systemen wie Servern oder Mainframes ist heute eine Standardfunktion aller Serversysteme und umfasst alle physikalischen Bestandteile eines entsprechenden Systems.

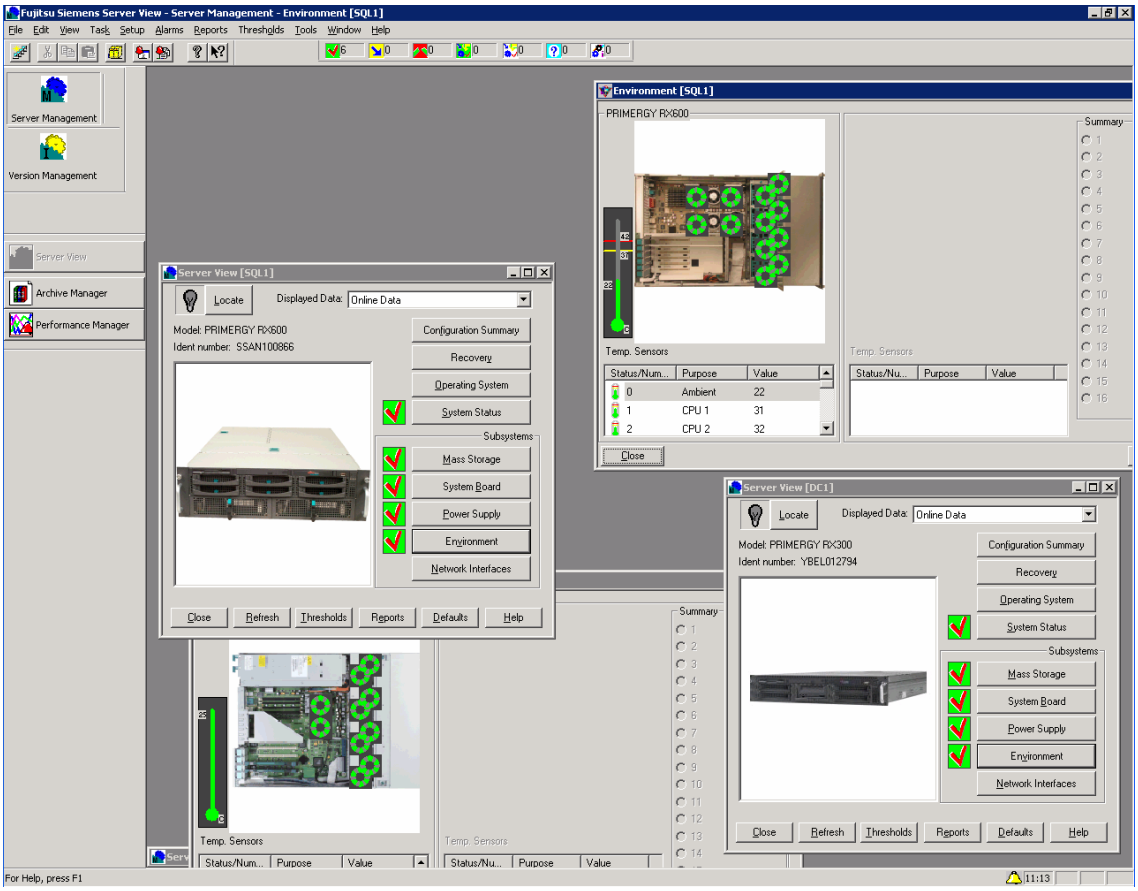

Abbildung 13 – FSC Serverview

Dazu gehören nicht nur der Server selbst, sondern auch angeschlossene Speichersubsysteme, Netzwerkverbindungen und SAN Verbindungen. Wichtige Tools in diesem Umfeld sind die Ma-

nagementsysteme der verschiedenen Hersteller wie FSC ServerView, Dell OpenManage oder die breiter etablierten Systeme wie Tivoli und HP/Compaq Insight Manager, die auch in der Lage sind, Systeme von Fremdherstellern zu überwachen. Die Daten dieser Managementwerkzeuge können hier auch an Systeme weitergegeben werden, die auf der Ebene der Softwarekomponenten weitere Dienste überwachen und so zu einem Monitoring von IT-Diensten auf den Ebenen Software und Hardware benutzt werden.

Die Bereitstellung von Servern erfolgt heute meist noch durch das physikalische Hinzufügen von Servern in ein Rack oder ein Bladecenter. Durch entsprechende Provisioning-Mechanismen wie Microsoft RIS/ADS oder andere Tools wird anschließend ein Betriebsystem auf den entsprechenden Server gebracht[14]. Bei virtualisierten Infrastrukturen kann dies auch eine sehr leichtgewichtige Abstraktionsschicht sein, in der die virtuellen Maschinen ausgeführt werden[15].

Konzepte wie das bei BMW umgesetzte Storage On-Demand zeigen, wie sich entsprechende Prozesse weiterentwickeln können [BMW03]. In diesem Szenario werden von BMW Speicherbereiche basierend auf Dienstgüte gekauft. Die Speicherinfrastruktur befindet sich zwar physikalisch im BMW Rechenzentrum und steht auch wie bisher direkt mit den Servern verbunden zur Verfügung, die notwendigen Hardwarekomponenten werden aber nicht von BMW sondern von HP-Mitarbeitern betrieben. Verrechnet wird nach tatsächlichem Verbrauch, die Hardware gehört HP. Entsprechende Konzepte bietet beispielsweise auch IBM mit On-Demand Servern. Der Mainframe wird dabei physikalisch besser ausgestattet, als der Kunde ihn bestellt hat und das System so im Rechenzentrum des Kunden installiert. Benötigt der Kunde zu einem späteren Zeitpunkt mehr Rechenleistung, werden zusätzliche Prozessoren einfach durch Software freigeschalten. Dies geht soweit, dass selbst ein Freischalten nur für einige Wochen möglich ist. Neue Angebote wie das SUN Grid, bei dem die Rechenleistung sogar Stundenweise über das Internet gemietet werden kann zeigen, dass IT immer mehr zu einem Dienst wie Strom oder Wasser wird. Bezahlt wird der Dienst nach Verbrauch, bereitgestellt direkt bei Nachfrage, auf möglichst einfache und für den Kunden transparente Weise. Die Art und Weise, wie der Dienst erzeugt wird, ist für den Kunden unerheblich, wie zum Beispiel bei Strom ist es irrelevant, wie dieser erzeugt wurde. Darüber hinaus muss der Kunde nicht in eine Infrastruktur investieren und Kapazitäten bereithalten.

Management von Softwarekomponenten

Aufbauend auf den Funktionen der Überwachung der reinen Hardwarefunktionen ist die Überwachung und das Management von Software auf den verschiedenen Systemen die nächste Stufe bei der Verwaltung von IT-Diensten.

Dabei müssen verschiedene Dienstebenen unterschieden werden. Elementare Funktionen stellt das Betriebsystem zur Verfügung, zum Monitoring auf dieser Ebene bieten Systeme wie Microsoft Operations Manager (MOM) oder Tivoli Funktionen, um die Basisdaten eines Servers wie die Hauptspeicherauslastung, sowie den Plattenplatzverbrauch und andere Messpunkte zu überwachen und zu analysieren. Darüber hinaus wird auf dieser Ebene auch das Verhalten des Betriebsystems analysiert. Die Analyse entsprechender Daten, sowie das Wissen über die Zusammenhänge der Messwerte wird in MOM dabei in so genannten Management Packs hinterlegt. Diese beinhalten die Logik der jeweils zu überwachenden Applikation oder des Betriebsystems. Sie realisieren dabei das Wissen über den IT-Dienst, seine verschiedenen möglichen Konfigurationen und die Abhängigkeiten von anderen Systemen und stellen dabei einen Teil unseres Modells in abstrakter Form dar. Ein Beispiel dafür ist das Management Pack für Active Directory (AD) von MOM, dem zentralen Verzeichnisdienst einer Windows Server Infrastruktur. Das Active Directory dient zur Verwaltung aller Informationen über die Benutzer und die Server in einer Windows Infrastruktur. Im AD werden darüber hinaus auch die Sicherheitsrichtlinien festgelegt. Der Dienst wird aus verschiedenen Domänencontrollern aufgebaut, die eine verteilte

[14] Oft wird hier auch von Bare-Metal Deployment gesprochen, da auf einen Server ohne jede Information ein lauffähiges System gebracht wird

[15] Ein Beispiel hierfür ist VMWare ESX Server

Datenbank mit Replikationsmechanismen implementieren. Das Management Pack überwacht alle Ereignisse, die das Active Directory auf den verschiedenen Servern erzeugt. Mit dem Wissen über die Grundstruktur des Active Directory und den Daten, die das Management Pack über die vorliegende Installation gesammelt hat, kann das Management Pack Fehler erkennen, sowie deren Auswirkung auf den gesamten, verteilten Dienst bestimmen. Das Wissen über diese Struktur und die Verteilung dieses speziellen Dienstes ist im Management Pack hinterlegt.

Eine weitere Aufgabe auf der Ebene der Softwarekomponenten ist die Bereitstellung von Software auf den Systemen. Dabei können verschiedene Ebenen unterschiedenen werden. Die Bereitstellung von Software auf den Systemen erfolgt durch Produkte wie den Microsoft Systems Management Server (SMS)[16]. Dabei werden Applikationen auf das jeweilige System kopiert und dort installiert. Diese Systeme eigenen sich auch für ein Patchmanagement von Systemen und sind in der Lage, die Verteilung von Software auf den verwalteten Systemen zu prüfen. Dabei ist eine Integration zwischen den verschiedenen Ebenen zu beobachten. IBM und Dell bieten seit kurzem Systeme an, mit denen sich mit SMS die BIOS-Daten ihrer Server abfragen und neue BIOS Versionen verteilen lassen. Gleiches gilt für Hardwaretreiber. Proprietäre Systeme der Hersteller werden dabei überflüssig und die erweiterten Funktionen auf der Softwareebene werden für diese Systeme nutzbar. Denkbar sind hier BIOS-Updates abhängig von den installierten Softwarekomponenten des Servers, zum Beispiel die Installation einer speziellen BIOS Version vor der Installation eines Betriebsystemupdates auf einem speziellen Server.

Programme wie Application Center 2000 oder die Funktionen in J2EE Containern wie BEA Weblogic setzten auf der Ebene der Applikationen an. Diese Systeme ermöglichen es, die Daten einer Anwendung auf mehreren Systemen synchron zu halten. Entsprechende System werden vor allem bei Clustersystemen eingesetzt und ihre Funktionen beschränken sich auf eine spezielle Applikation. Darauf aufbauend können Systeme wie IBM Orchestration auch die Verteilung von Komponenten, auf deren Management sie ausgelegt sind, auf verschiedenen Systemen dynamisch ändern und so auf geänderte Anforderungen oder Lastspitzen reagieren. Ein weiteres Beispiel dafür ist SUN N1 Grid für SAP oder Fujitsu Siemens Adaptive Services Control Center (ASCC).

Management der Netzinfrastruktur

Das Management der Infrastruktur erfasst Komponenten wie Netzwerkgeräte oder SAN Infrastrukturen. Dabei werden Switches und ihre Verbindungen durch Systeme wie HP ProCurve verwaltet. Diese Systeme dienten anfangs lediglich dazu, die Hardware wie Switches an sich zu verwalten. Neue Entwicklungen zeigen, dass immer mehr Funktionen in entsprechende Produkte integriert werden, durch die sich neuartige Funktionen für die Netzwerkinfrastruktur ergeben. Im Bereich der Sicherheit sind hier vor allem HP ProCurve Identity driven Networking und die entsprechenden Bemühungen von Cisco im Bereich Network Access Protection zu nennen. Damit wird es möglich, physikalische Netzwerkports eines Ethernet-Switches basierend auf Softwareregeln zu verwalten. So kann ein Rechner zum Beispiel von einem VLAN in ein anderes verschoben werden, wenn die Managementinfrastruktur einen Virus auf dem Rechner erkennt, um die Ausbreitung des Virus im Unternehmen zu verhindern.

Diese Systeme sind heute meist noch losgelöst und nicht in ein vollständiges Infrastrukturmanagement integriert. Meist beschränkt sich die Integration auf das Weitermelden von Statusinformationen. Durch die Integration dieser Fähigkeiten in neue Produkte wie den Windows 2008 Server von Microsoft werden diese Funktionen aber in Zukunft immer häufiger in Netzwerken anzutreffen sein, weil sie einen erheblichen Beitrag zur Absicherung des Netzwerkes leisten können.

Management von Geschäftsprozessen

Eine neue Managementebene im Bereich der Softwaresysteme ist durch Applikationen wie Biztalk und SAP XI entstanden. Biztalk führt Geschäftsprozesse aus, die in BPEL modelliert sind. Diese XML basierte Sprache erlaubt es, Geschäftsprozesse nicht in Applikationslogik,

[16] Mittlerweile Microsoft System Center Configuration Manager

sondern in XML-Dokumenten abzulegen. Dadurch können diese Geschäftsprozesse deutlich einfacher geändert werden. Darüber hinaus ermöglicht BPEL so dem Fachverantwortlichen die Implementierung des Geschäftsprozesses, da die Beherrschung einer Programmiersprache nicht mehr notwendig ist. Wir gehen in einem der nächsten Abschnitte noch genauer auf BPEL ein. Der Container wie Biztalk führt diese Geschäftsprozesse dann aus und ermöglicht auch die Überwachung. Durch dieses so genannte Business Activity Monitoring (BAM) ist es möglich, den Zustand von Geschäftsprozessen zu überwachen. Dieses Monitoring ist aber auf die Geschäftsprozesse innerhalb des Containers und seiner Schnittstellen beschränkt und umfasst weder die darunter liegende Software, noch die Hardware auf der das System ausgeführt wird.

Entscheidend wird in diesem Zusammenhang das End-to-End Monitoring. Wichtig ist nicht nur, ob ein Dienst auf einem Server verfügbar ist, sondern auch, ob der Benutzer auf diesen Dienst zugreifen kann und der Dienst auch fachlich korrekt arbeitet.

Tools zur Unterstützung der ITIL Prozesse

Neben den Managementfunktionen auf den einzelnen Ebenen haben sich auch Tools rund um die ITIL Prozesse herausgebildet. Wir stellen hier die Tools vor, die für das Configuration Management relevant sind. Auch die bereits beschriebenen Produkte tragen aber ihren Teil zu den ITIL Tools bei. Im Markt lässt sich eine deutliche Standardisierung erkennen, die Tools wachsen immer mehr aufeinander zu. Gerade die großen Hersteller integrieren dabei neue Funktionen durch den Aufkauf kleinerer Firmen in ihre Produktportofolios und integrieren diese Funktionen dann in ihre jeweiligen Managementplattformen.

Configuration Management Database

Wir haben die Configuration Management Database (CMDB) im Abschnitt über das Configuration Management bereits kurz eingeführt. Die CMDB realisiert den Datenspeicher für das ITIL Configuration Management. Dazu müssen die Informationen und Attribute der verschiedenen Configuration Items (die CI's) abgelegt werden. Darüber hinaus werden die Verbindungen zwischen ihnen gespeichert. Eine CMDB ist damit der zentrale Datenspeicher für alle CI's. Dabei ist das Schema einer CMDB oft auf dem CIM Standard abgestützt, wird jedoch meist von den einzelnen Herstellern erweitert. Darüber hinaus wird dieses Schema auch sehr oft an die Bedürfnisse des jeweiligen Kunden angepasst.

Betrachten wir unseren Prozess für existierende Infrastrukturen, so findet sich die CMDB als der Informationsspeicher des Gesamtmodells (siehe Abbildung 14). Die CMDB an sich ist ein Datenspeicher, der angelegt und gepflegt werden muss. Dabei müssen bei der Einführung einer CMDB die Informationen über bereits vorhandene Configuration Items in der CMDB erfasst und Änderungen nach ihrer Durchführung eingepflegt werden.

Wir betrachten als Beispiel für eine CMDB die CMDB Atrium von BMC/Remedy. Die Datenbank lässt sich allein stehend oder zusammen mit den anderen ITIL Tools von BMC einsetzen. Zwei wichtige Tools für die CMDB sind dabei:

- BMC IT Discovery Suite
- Asset Management Application

Die BMC IT Discovery Suite ist eine Sammlung von Application Scannern, die die Verteilung von Anwendungen auf verschiedenen IT-Systemen in der IT-Infrastruktur erkennen. Diese Suite nimmt damit die Funktion des Application Scanners war und kann so den IST-Zustand der IT-Dienste erfassen. Wir werden die verschiedenen Aufgaben eines Application Scanners im nächsten Abschnitt beschreiben.

Die BMC Asset Management Application ist das Produkt zur Inventarisierung und Verwaltung der IT-Infrastruktur mit ihren Hardwarekomponenten. Diese basiert auf der Atrium CMDB und stellt so die Schnittstelle zur Inventarverwaltung von der CMDB hin zu einem verantwortlichen Mitarbeiter dar. Hier zeigt sich, dass sich heutige CMDB Systeme sehr häufig aus Applikationen zur Inventarisierung der IT-Infrastruktur entwickelt haben.

Die Abbildung 15 verdeutlicht, welche Funktionen und Prozesse auf die CMDB zugreifen.

Die wichtigsten Funktionen dabei sind:

- Überblick über die CI's in der CMDB für die verschiedenen Benutzer und Verantwortlichen der verschiedenen ITIL-Prozesse
- Gemeinsames Datenmodell für alle Tools, die Funktionen beim Management der IT-Infrastruktur ausführen, durch die Etablierung eines einheitlichen Datenmodells
- Reconciliation Engine zur Integration von CI-Elementen und Attributen aus anderen Quellen in die CMDB
- Offene Schnittstellendefinitionen zu anderen Systemen

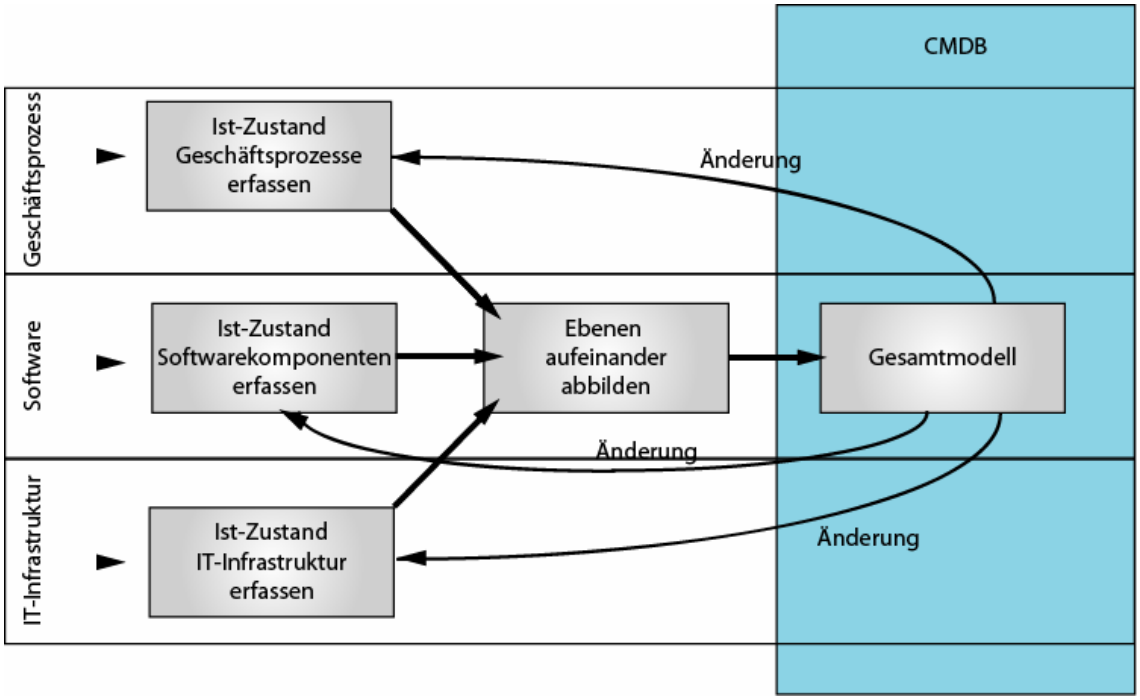

Abbildung 14 – CMDB im Prozess existierende Infrastruktur

Zur Atrium CMDB gehören weitere Produkte, die verschiedene ITIL Prozesse realisieren und auf der CMDB basieren:

- Service Desk, als Ticketsystem zur Verwaltung von Supportanfragen und als Schnittstelle zwischen Anwender und Change Management Tool
- Change Management, als Tool zur Verwaltung von Änderungen
- Asset Management, als Tool zur Inventarisierung der IT-Infrastruktur und der eingesetzten Software
- Service Impact Manager, als Tool zur Planung von Änderungen und deren Auswirkungen auf die Infrastruktur

Die mit Atrium CMDB beschriebene Realisierung stellt eine einzige CMDB innerhalb des Unternehmens dar, die alle Informationen über die gesamte Infrastruktur enthält und mit der alle eingesetzten Werkzeuge zusammenarbeiten. Dies ist jedoch meist nicht möglich, da fast immer Werkzeuge unterschiedlicher Hersteller eingesetzt werden, die jeweils ihre eigenen Datenbanken beinhalten. Ein Beispiel dafür ist der im nächsten Abschnitt betrachtete Application Scanner MAM, der eine eigene CMDB beinhaltet. Diese Notwendigkeit zeigt sich auch daran, dass die

meisten CMDBs Konzepte beinhalten, wie sich Informationen aus anderen Datenbanken integrieren lassen. Im Falle von Atrium übernimmt diese Aufgabe die Reconciliation Engine.

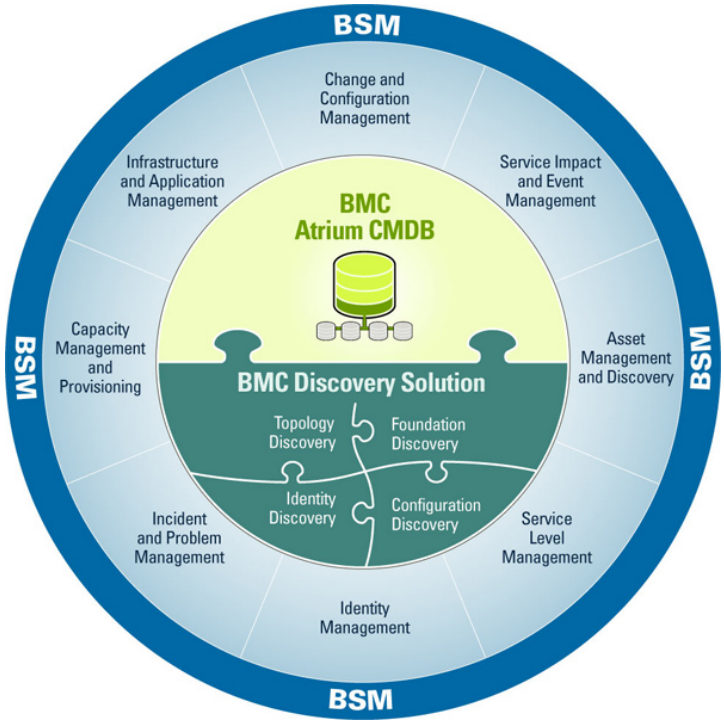

Abbildung 15 – Einbettung einer CMDB am Beispiel BMC Atrium (Quelle: www.bmc.com)

Es zeigt sich darüber hinaus, dass die Grenzen zwischen den ITIL Prozessen teilweise sehr fließend sind. Die CMDB nimmt auch Aufgaben von anderen Prozessen war, die keine Kernaufgaben des Configuration Managements sind und spielt gerade für das Incident Management eine wichtige Rolle.

Die Aufgaben eines Application/Topology Scanners

Ein Application Scanner analysiert die IT-Infrastruktur, erkennt die Verteilung und Konfiguration von Anwendungen auf IT-Systemen und analysiert die Verbindungen zwischen den Systemen. Darüber hinaus kann er eine Inventarisierung der Hardwarekomponenten vornehmen. Er nimmt damit eine wichtige Aufgabe im Change-Management Prozess war, indem er die Informationen über die Infrastruktur in die CMDB übernimmt. In einem Bottom-Up orientierten Erstellungsprozess für ein IT-Modell ist diese Komponente der Ausgangspunkt für die Erstellung des Modells des IT-Dienstes.

Verwendet wird für einen Application Scanner auch der Begriff des Topology Scanners, dieser Begriff wird aber auch teilweise für Anwendungen benutzt, die nur eine Netzwerkinfrastruktur scannen und die physikalische Struktur eines Netzes darstellen. Ein Beispiel für diese Art von Anwendungen wäre der bereits früher vorgestellte HP ProCurve Manager. Wir sprechen im Weiteren von Application Scannern und beziehen uns damit auf Werkzeuge, die neben den Hardwareinformationen der IT-Infrastruktur auch Softwarekomponenten auf den Servern erkennen können.

Dazu kann ein Application Scanner unterschiedliche Techniken einsetzen:

- Abhören des Netzwerkverkehrs in einem Rechenzentrum, um Kommunikationsmuster zu erkennen
- Analyse der Netzwerkstruktur durch Senden von Nachrichten (ping, SNMP Traps)

- Prüfen einzelner Server von außen und testen, welche Services darauf ausgeführt werden
- Inventarisierung der Hardware
- Installation eines Agenten auf dem Server und Analyse der offenen Verbindungen nach außen mit Prüfung, welche Programme diese Ports öffnen
- Analyse der installierten Anwendungen auf dem Server
- Analyse von Konfigurationsdateien und Logbüchern von Anwendungen auf dem Server

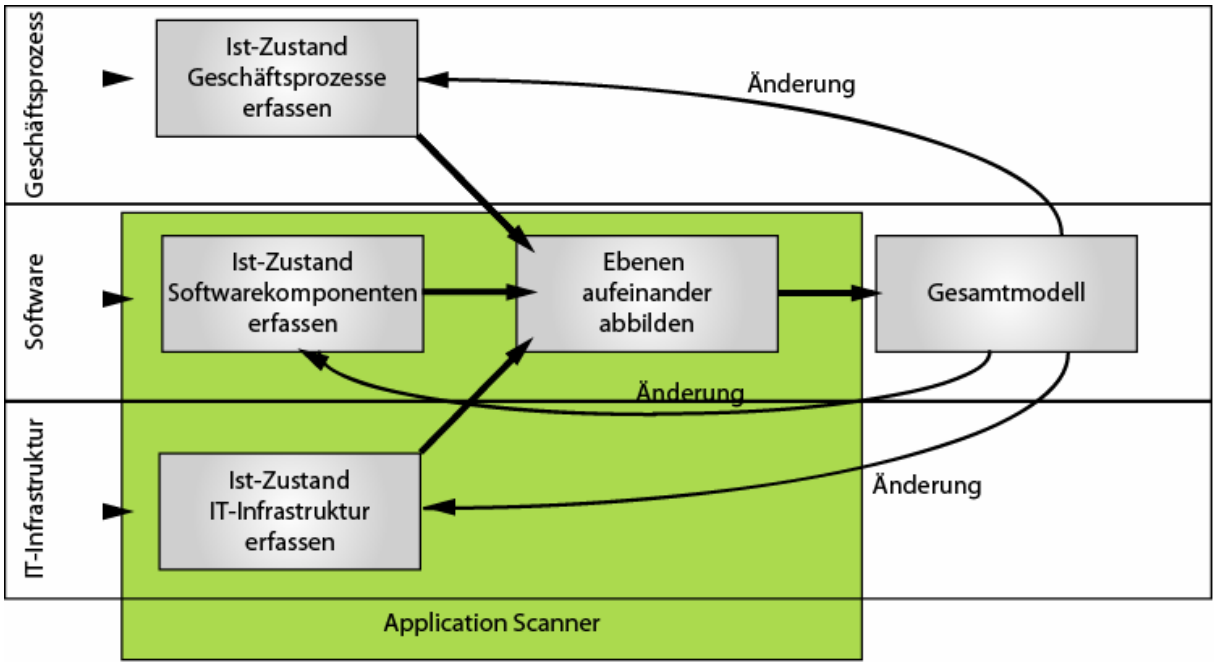

Abbildung 16 – Application Scanner Prozess existierende Infrastruktur

Diese Techniken stellen eine mögliche Hierarchie dar und können entsprechend erweitert werden. Je höher der Scanner in der Hierarchie arbeitet, desto mehr Informationen und Zugriffsrechte braucht er auf die einzelnen Komponenten. Die Informationen zur Erkennung lassen sich zum Beispiel durch synthetische Tests in einer Entwicklungsumgebung generieren. Durch das Ausführen einer Anwendung und Analyse der Aufrufe zwischen den Komponenten lassen sich Fingerabdrücke von Applikationsaufrufen erstellen, die dann als Fingerabdrücke für den Application Scanner dienen können und anhand derer er Kommunikationsmuster in einem Netzwerk erkennen kann[17]. Diese Fingerabdrücke beschreiben dabei zum einen die eine Anwendung und Möglichkeiten zur Erkennungen, in den Fingerabdrücken können aber auch die Beziehungen zwischen Anwendungen abgelegt werden. Die Abbildung 17 zeigt einen entsprechenden Fingerabdruck für Mercury Application Mapping (MAM), den Application Scanner der Firma Mercury. Zur Beschreibung der Abhängigkeiten verwendet MAM eine Sprache, die Topology Query Language (TQL). TQL basiert auf der Structured Query Language (SQL), die zur Abfrage von Datenbanken verwendet wird, und erweitert SQL um eine grafische Darstellungsform und die Modellierung von verschiedenen Beziehungen[18]. In TQL lassen sich unter anderem Finge-

[17] Wichtig ist dabei, dass kein „Hintergrundrauschen“ in die Fingerabdrücke gelangt, wenn andere Applikationen oder Betriebsystemprozesse gleichzeitig ausgeführt werden.

[18] Dies ist vergleichbar mit Relationen innerhalb einer Datenbank, die um Attribute erweitert werden.

rabdrücke von Applikationen und IT-Infrastrukturkomponenten erstellen, die das System in der Infrastruktur wieder erkennen kann. Die Abbildung 17 stellt eine Anwendung als Beispiel dar.

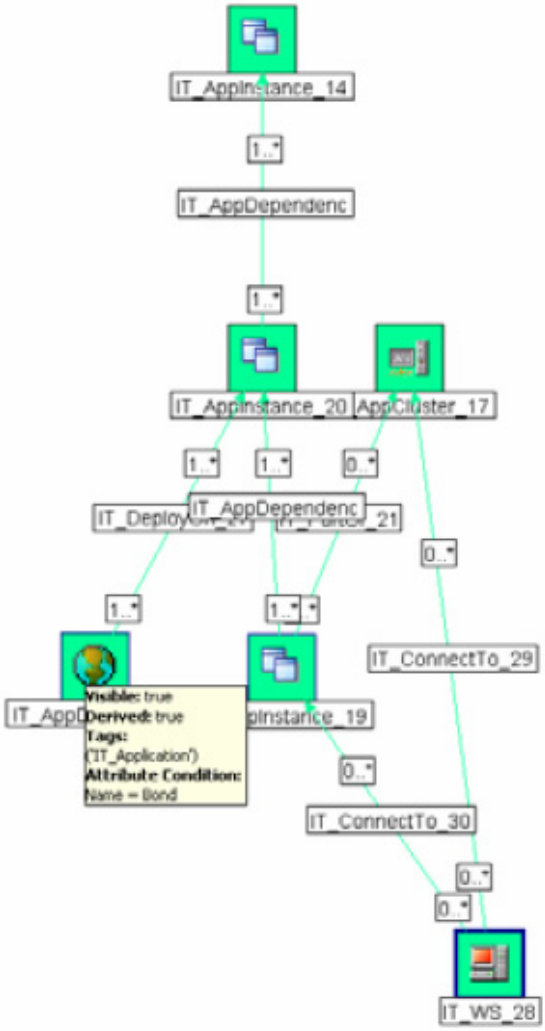

Abbildung 17 – Fingerabdruck einer Applikation in TQL (Quelle Mercury)

Wenn man die Funktionen eines Application Scanners wie in Abbildung 16 dargestellt einordnet, zeigt sich, dass dieser vor allem den Prozess der Inventarisierung der IT-Landschaft übernimmt. Dabei sind zwei verschiedene Situationen zu unterscheiden:

- Der erste Scanlauf zur Erfassung des Zustands und der Verteilung der IT-Komponenten, der die IT-Infrastruktur analysiert und die Grundstruktur in der CMDB erstellt.
- Scanläufe, die in regelmäßigen Abständen die IT-Infrastruktur analysieren und so Veränderungen in die CMDB spiegeln. Dabei können dadurch sowohl Change-Request abgeschlossen und in die CMDB eingepflegt werden, als auch unbeabsichtigte (oder nicht freigegebene) Veränderungen erkannt werden.

Ein Application Scanner sollte in beiden Situationen eingesetzt werden können. Untersuchungen von Mercury gehen davon aus, dass sich die Kosten für die Erfassung und Pflege der IT-Infrastruktur und der Software in der CMDB durch den Einsatz eines Application Scanners um mehr als 70% verringern lassen. [Mercury06-2].

Fazit

Für verschiedene Aufgaben, gerade auch für die ITIL Prozesse, stehen unterschiedliche Werkzeuge zur Verfügung. Dabei kommen die Werkzeuge aus unterschiedlichen Bereichen und wachsen immer weiter aufeinander zu. Es gibt momentan kein Tool, das alle Dienste eines Rechenzentrums so steuern könnte, wie wir in unserer Vision beschrieben haben. Die verschiedenen Tools verwenden darüber hinaus unterschiedliche Datenbasen, auf denen sie operieren. Die Ansätze, mit einer CMDB eine Gesamtdatenbank mit allen Informationen zu generieren, führen zu sehr großen Datawarehouses, die nur sehr schwer zu managen und pflegen sind.

Wir werden im nächsten Abschnitt einen anderen Ansatz zur Erstellung des Modells eines IT-Dienstes vorstellen, bevor wir auf die technischen und organisatorischen Herausforderungen, der sich die IT heute gegenübersieht, eingehen.

2.3.4 Standards zur Modellierung

Wir haben im vorigen Abschnitt Tools vorgestellt, die die Modellierung von IT-Diensten auf einzelnen Ebenen ermöglichen. Es gibt heute keine Modellierungstechnik, die eine feingranulare Modellierung eines IT-Dienstes auf allen drei Ebenen ermöglicht. Es gibt jedoch einige Standards, die wir in diesem Abschnitt kurz vorstellen möchten. Wir geben hier einen Überblick über die am weitesten verbreiteten Techniken auf den verschiedenen Ebenen. Dieser Überblick ist aber auf keinen Fall vollständig und stellt nur eine kurze Einführung dar.

ARIS

Die IDS Scheer AG hat mit dem ARIS Toolset eine mächtige Modellierungstechnik entwickelt, die sich nicht nur zur Modellierung von Geschäftsprozessen sondern auch zu deren Umsetzung in Betriebliche Informationssysteme wie SAP Netweaver und Workflow-Managementsystemen eignet[19]. ARIS wird heute in vielen großen Konzernen zur Geschäftsprozessmodellierung erfolgreich eingesetzt, unter anderem bei der Siemens AG.

Abbildung 18 – ARIS Haus (Quelle [AR01])

Das ARIS Toolset erlaubt mit dem HOBE[20] Ansatz sowohl die Modellierung der Geschäftsprozesse als auch die Modellierung der Netzwerkkonfiguration und der Softwaremodule. Dabei stützt sich ARIS auf Modellierungstechniken wie EPKs, der UML und Relationaler Modellierung von Datenbanken ab und integriert diese in eigene Darstellungen, so genannten Sichten. Die

[19] ARIS for Netweaver

[20] ARIS HOBE – ARIS **H**ouse **o**f **B**usiness **E**ngineering

Modellierung der Geschäftsprozesse, sowie der verwendeten Software und der IT-Infrastruktur erfolgt in ARIS mit entsprechend vorhandenen Werkzeugen [AR01].

Die Modellierung von Geschäftsprozessen in ARIS ist nicht nur auf der Ebene von Geschäftsprozessen als Ganzes möglich, durch die Ereignisprozessketten (EPK) können auch die verschiedenen Abläufe eines Geschäftsprozesses modelliert werden. Dies umfasst neben den Abläufen, die auf IT-Systemen umgesetzt werden auch Abläufe, die von Akteuren, also Personen durchgeführt werden. Die Abbildung 19 gibt ein Beispiel für eine erweiterte Ereignisprozesskette (eEPK) an. Durch eEPK's werden den EPK's Informationen über die Organisations-, Daten und Leistungsmodellierung hinzugefügt.

Die Modellierung von Software in ARIS orientiert sich an der Objektorientierten Programmierung. Softwarekomponenten sind dabei Module, die im Sinne einer objektorientierten Verfeinerung auch verschachtelt sein können. Dabei fokussiert ARIS sehr stark auf die Anforderungen des Entwicklers und dessen Darstellungen. Informationen über die innere Struktur sind für den Betrieb aber meist unerheblich.

Die Abbildung von Diensten im Sinne einer Service orientierten Architektur auf ARIS Module ist zwar möglich, sie gibt aber nicht die gesamte Mächtigkeit entsprechender Ansätze wieder. Die (meist) statischen Beziehungen in objektorientierten Anwendungen werden durch dienstbasierte Architekturen wie Webservices dynamischer. Durch die Integration von entsprechenden Schnittstellen in Standardanwendungen wie zum Beispiel Microsoft Office wird es damit auch Endanwendern ermöglicht, verteilte Anwendungen zu erstellen. Gerade Microsoft Excel bietet hier mächtige Möglichkeiten zur Datenanalyse, an die externe Datenquellen leicht angebunden werde können. Dabei können Anwendungssysteme entstehen, die nicht von einem Entwickler entworfen wurden und die nicht zentral geplant und gesteuert werden. Die Integration entsprechender Anwendungen in ARIS Modelle ist schwierig, da eine zentrale Bereitstellung dieser Anwendungen meist nicht erfolgt und die Existenz dieser Anwendungen und Beziehungen nicht sichtbar wird.

Die Modellierung von Netzwerken und IT-Diensten in ARIS basiert auf den Konzepten des Asset Managements, in ARIS auch Betriebsmittelverwaltung bezeichnet. Die Ansätze sind dabei teilweise sehr abstrakt gehalten und enthalten so nicht alle Informationen über die Netzwerkstruktur. Die Modellierung von Firewallstrukturen und logischen Infrastrukturen wie virtuellen Netzen ist dabei nur schwer möglich.

Auch die Abbildung von Anforderungen der Softwarekomponenten an die Infrastruktur, die Abbildung von Ausführungsbeziehungen wie J2EE Anwendungsserver, ist in ARIS nur als manuell gepflegte Beziehung möglich. Ähnliches gilt für virtualisierte Infrastrukturen und die neuen Möglichkeiten zur Sicherung von virtuellen Kanälen in Servern.

Die Modellierung von Anwendungsservern und Betriebsystemen erfolgt in ARIS nur in den Hardwarekomponenten [AR01-S66]. Andererseits werden Informationen in ARIS abgelegt, die für die Ausführung nicht relevant sind. Konzepte wie Standorte und zugeordnete Organisationsstrukturen sind in einer virtuellen IT-Infrastruktur nicht entscheidend für den Zugriff auf IT-Dienste und entstammen wieder den Konzepten und Datenmodellen des Asset Managements.

Gerade die Entwicklungen des letzten Jahres rund um ARIS zeigen, dass sich die verschiedenen Ansätze aufeinander zu bewegen und der Weg hin zu einem integrierten Management von IT-Diensten von vielen gegangen wird. Die Ansätze dazu unterscheiden sich. ARIS hat als Modellierungswerkzeug mit dem ARIS Toolset hier den Vorteil, dass es für die Geschäftsprozessmodellierung bereits etabliert ist. Die Integration in Standardlösungen für betriebliche Informati onssysteme, wie ARIS for Netweaver ermöglicht es, die Ebene der Geschäftsprozesse mit Standardanwendungen einfach zu verbinden. Dies erfolgt aber nur automatisiert für die Netweaver-Komponenten, die ARIS for Netweaver kennt. Für selbstentwickelte Softwarekomponenten und Module muss die Abbildung auch weiterhin durch den Administrator in Zusammenarbeit mit dem Softwareentwickler erfolgen.

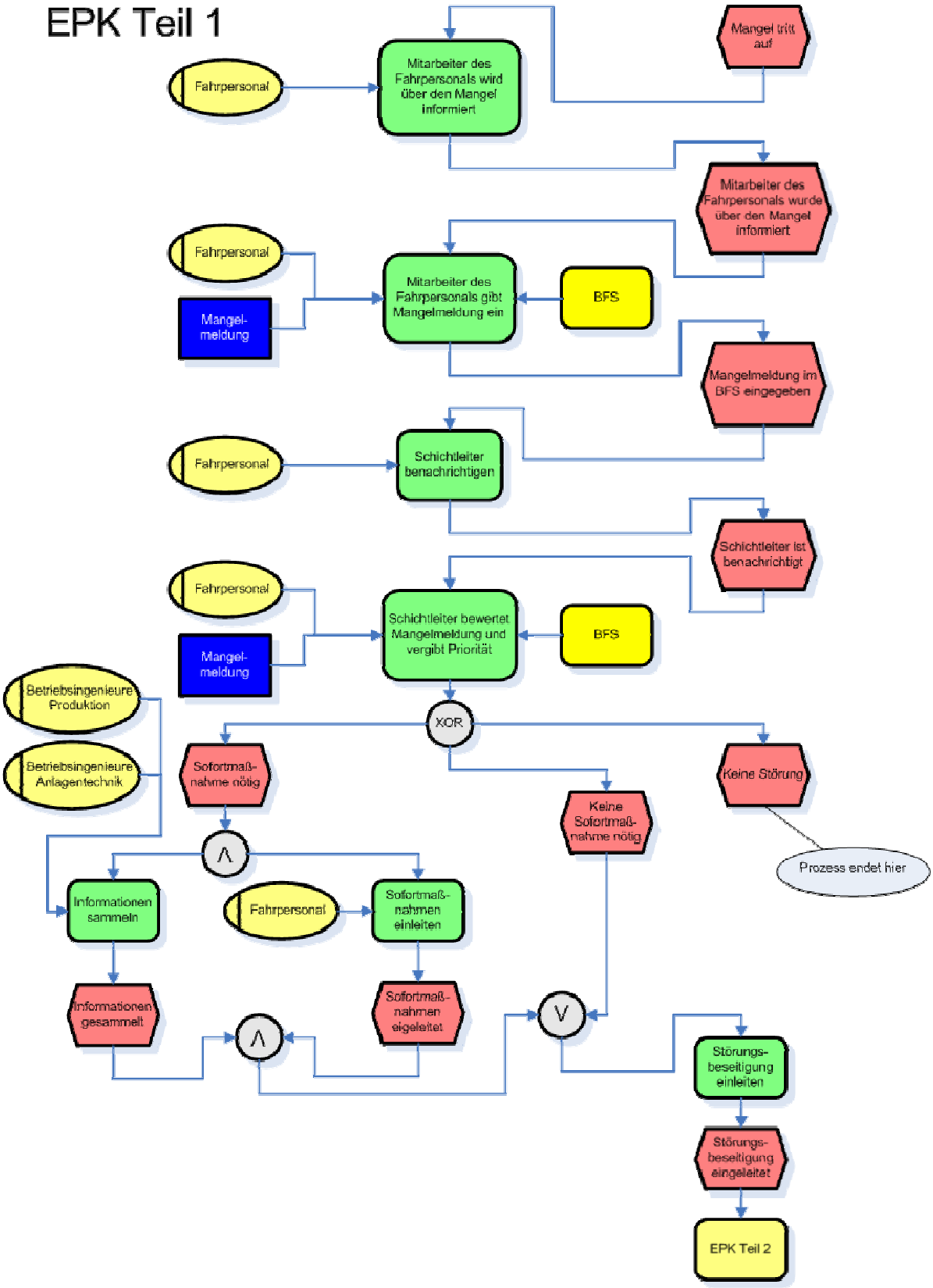

Abbildung 19 – Komplexes eEPK (Quelle Wikipedia)

Business Process Execution Language - BPEL

Im Gegensatz zu ARIS bewegt sich die Business Process Execution Language (BPEL) zwischen den Ebenen Geschäftsprozessmodellierung und Softwareentwicklung. BPEL ist ein Standard der Organization for the Advancement of Structured Information Standards (OASIS) und liegt miterweile in Version 2.0 vor [OABPEL]. Viele Firmen, darunter die Initiatoren IBM, Microsoft und BEA unterstützen BPEL durch Produkte und Integrationsmöglichkeiten in ihre Softwarestacks.

BPEL erlaubt ein so genanntes „Programmieren im Großen“. Dazu werden Prozesse verwendet. Die Prozesse kommunizieren dabei über Webservices.

> *„Ausführbare BPEL-Prozesse können auf einer Workflowmaschine zum Einsatz gebracht werden (engl. deployed) und sind durch sie ausführbar. Abstrakte Prozesse dienen der Beschreibung des Verhaltens des Prozesses ("behavioural interface"). Sie werden als Sicht auf einen ausführbaren Prozess verwendet und dienen dazu, das interne Verhalten des Prozesses z.B. vor einem Geschäftspartner zu verbergen.“*

Definition 10 – Prozesse in BPEL (Quelle [WP-BPEL])

Die Abläufe und die Interaktionen auf den fachlichen Objekten und Nachrichten wird in BPEL beschrieben. Man spricht hier auch von Webservice-Orchestrierung, weil mit BPEL die „Musik“, die die Webservices, das „Orchester“ spielen, beschrieben wird. Die Notwendigkeit zur Entwicklung der Webservices und der fachlichen Objekte bleibt weiterhin bestehen. Es ist jedoch durch BPEL einfach möglich, die Abläufe eines Prozesses anzupassen.

BPEL Anwendungen werden innerhalb einer BPEL Engine ausgeführt. Die Engine kümmert sich dabei darum, dass die Anwendungen geladen und verarbeitet werden, sie stellt die Laufzeitumgebung zur Verfügung, kümmert sich um die Sicherheit der Anwendungen und um die Basisdienste wie die Bereitstellung von Transaktionssicherheit. Ein Beispiel für eine entsprechende Engine ist der Microsoft Biztalk Server.

BPEL ersetzt weder die Softwareentwicklung, noch wird damit die Geschäftsprozessmodellierung ersetzt. Es stellt aber eine Schicht dar, die sowohl der Entwickler als auch der Fachverantwortliche gemeinsamen anpassen können und die dazu beiträgt, dass die Kluft zwischen den beiden Ebenen etwas schmaler wird.

UML

Die wohl wichtigste und am meisten verbreitete Modellierungssprache für Softwaresysteme ist heute die Unified Modelling Language (UML). Sie wird von der Object Management Group (OMG) standardisiert.

> *„Die Unified Modelling Language (UML) ist eine von der Object Management Group (OMG) [OMG] entwickelte und standardisierte Sprache für die Modellierung von Software und anderen Systemen. Im Sinne einer Sprache definiert die UML dabei Bezeichner für die meisten Begriffe, die für die Modellierung wichtig sind, und legt mögliche Beziehungen zwischen diesen Begriffen fest. Die UML definiert weiter grafische Notationen für diese Begriffe und für Modelle von statischen Strukturen und von dynamischen Abläufen, die man mit diesen Begriffen formulieren kann. Die UML ist heute eine der dominierenden Sprachen für die Modellierung von betrieblichen Anwendungssystemen (Softwaresystemen)“. [WP-UML]*

Definition 11 – Definition UML

Durch verschiedene Arten von Diagrammen werden dabei Softwarekomponenten und Applikationen modelliert und beschrieben. Dafür steht eine grafische Notation zur Verfügung. Die bekanntesten und für das Szenario wichtigsten Diagrammtypen sind:

- Use-Case Diagramme

  Use-Case Diagramm liefern Beschreibungen für einzelne Abläufe und zu implementierende Funktionen und werden vor allem während des Requirement Engineering eingesetzt.

- Sequenzdiagramme

  Sequenzdiagramme beschreiben die Abläufe und Aufrufbeziehungen zwischen verschiedenen Softwarekomponenten.

- Klassendiagramme

  Klassendiagramme beschreiben Softwarekomponenten mit ihren Eigenschaften und Methoden. Sie sind damit eine grafische Darstellung der Implementierung in einer Softwarekomponente.

- Paketdiagramme

  Paketdiagramme geben die Struktur einer Anwendung wieder und entsprechen damit der Darstellung von Anwendungen. Die Paketstrukturen geben meist die Grenzen zwischen den Anwendungen wieder.

- Deploymentdiagramme

  Deploymentdiagramme beschreiben die Verteilung von Artefakten auf Knoten, von Softwarekomponenten auf Hardware. Eine Modellierung der IT-Infrastruktur und der Beziehungen zwischen den Knoten erfolgt nicht.

Die UML richtet sich vor allem an Softwareentwickler. Sowohl eine detaillierte Modellierung von Geschäftsprozessen als auch eine detaillierte Modellierung der IT-Infrastruktur ist mit der UML nicht möglich. Durch die sehr gute Toolunterstützung und das so genannte „Round-Trip Engineering“, bei dem sowohl die Änderungen von Modellen im Code der Software als auch die Änderungen im Code in den Modellen synchronisiert werden, hat sich die UML als ein sehr gutes Mittel zur Modellbasierten Entwicklung von Software entwickelt.

Microsoft Dynamic System Initiative - DSI

Die Unterstützung von Microsoft für die UML ist nicht sehr stark ausgeprägt, mit Visual Studio 2005 steht für den Entwickler ein erstes Produkt der Dynamic Systems Initiative (DSI) zur Verfügung. Mit DSI will Microsoft ebenfalls den gesamten Lebenszyklus von Softwaresystemen abbilden. Ein Teil der DSI ist dabei die Beschreibung von IT-Infrastrukturen.

Die DSI und entsprechende IT-Modelle definieren Endpunkte, an die entweder Datenbanken, Fileservices oder Webservices gebunden werden können. Mit den generischen Endpunkten steht auch eine Möglichkeit zur Verfügung, andere Arten von Endpunkten zu integrieren. An diese Endpunkte kann der Entwickler anschließend Webservices (oder an die jeweiligen anderen Endpunkte entsprechende Dienste), die er entwickelt hat, binden. Die Endpunkte werden an IT-Systeme gebunden und beschreiben Eigenschaften des Containers, in dem sie ausgeführt werden. Ein Beispiel dafür ist bei einem Webservice der Dienst, der für die Anmeldung verwendet wird (Basic/Windows NTLM). Der Entwickler kann hier auch in seiner Entwicklungsumgebung Änderungen an diesem Server vornehmen. Zwischen den Endpunkten können Beziehungen modelliert werden, diese entsprechen dann den Aufrufbeziehungen zwischen den Komponenten.

Momentan erlaubt die DSI noch keine detaillierte Modellierung der IT-Infrastruktur. Die Modelle müssen momentan auch noch manuell im Visual Studio entworfen werden. Visual Studio 2005 ist das erste Produkt dieser Reihe. DSI wird auch in die nächste Version des System Management Server, die Version v4 integriert und wird auch ein wichtiger Bestandteil des Microsoft Systems Center werden. Damit steht der Microsoft Application Scanner sowie das Asset Management Tool als ein Teil des DSI Modells zur Verfügung. Die Integration zwischen Entwicklung und Betrieb wird an dieser Stelle deutlich verbessert, auch wenn einige Annahmen wie die direkte Interaktion des Entwicklers mit der Konfiguration des produktiven Servers ohne einen

Change-Management-Prozess etwas gewagt erscheint. Durch die Integration von BPEL und Biztalk auf der anderen Seite des Softwarestacks bietet Microsoft hier aber eine über die Entwicklung in den Betrieb reichendes Modell, das in Zukunft eine komplementäre Sicht zu ITIL im Bezug auf Entwicklung realisieren kann und damit einen wichtigen Beitrag zu einem ganzheitlichen IT-Management leisten kann.

CIM/WBEM

Das Common Information Model (CIM) ist ein von der Distributed Management Task Force (DMTF) entwickelter Standard zum Management von IT-Systemen. Durch CIM soll eine einheitliche Managementschnittstelle für das Management von IT-Systemen etabliert werden. CIM stellt dabei ein einheitliches, objektorientiertes Datenschema dar, das einen Grundstock an Informationen für eine Managementapplikation beschreibt und von den jeweiligen Herstellern noch erweitert werden kann. Ein Beispiel für eine entsprechende Implementierung ist die Windows Management Instrumentation (WMI), die CIM Implementierung von Microsoft in Windows. Durch CIM wird es möglich, dass verschiedene Managementsysteme ihre Daten untereinander austauschen. CIM erlaubt dabei eine sehr feingranulare Modellierung der IT-Systeme.

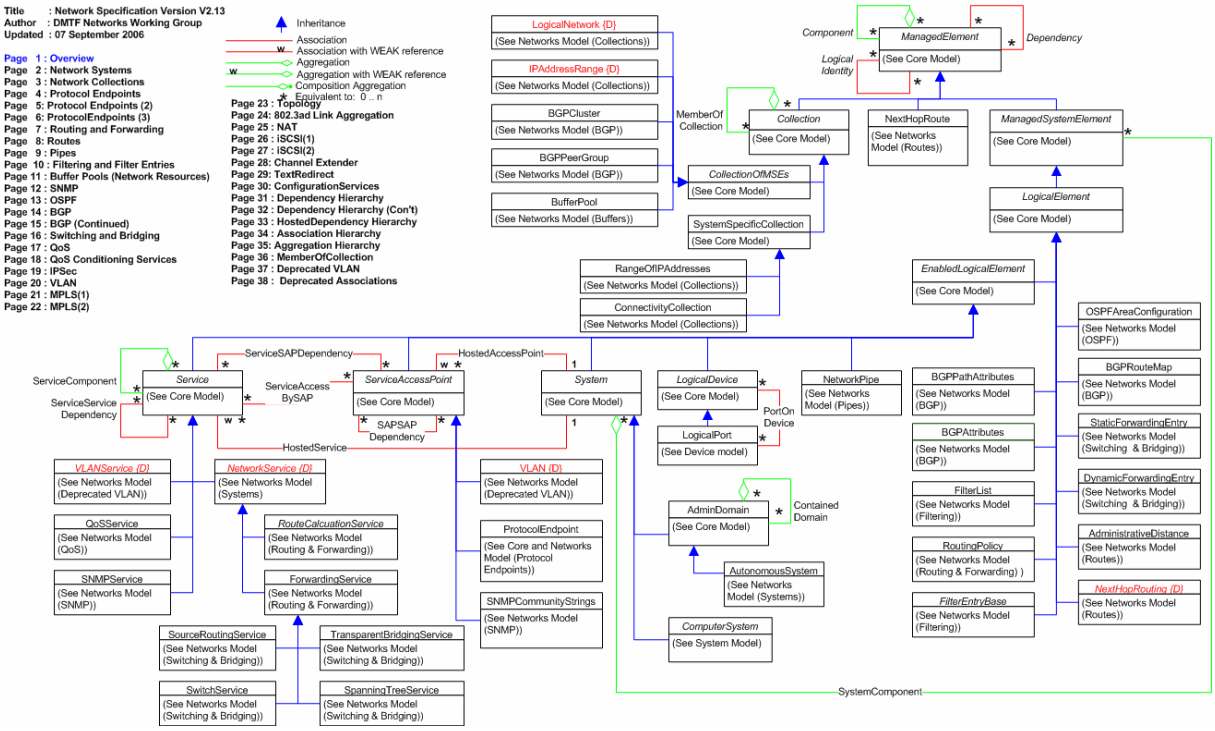

Abbildung 20 – Netzwerkmodellierung mit CIM (Quelle DMTF)

Die Abbildung 20 verdeutlicht dies am Beispiel der Modellierung auf der Netzwerkebene. Es stehen für verschiedene Netzwerkprotokolle verschiedene Endpunkte zur Verfügung, um die verschiedenen Kanäle und die verschiedenen Attribute der Endpunkte modellieren zu können.

Auch Web-Based Enterprise Management (WBEM) ist eine Initiative der DMTF. WBEM definiert den Zugriff auf die CIM Daten und ist kompatibel zu entsprechenden Standards wie SNMP der DMI. Damit wird es möglich, entfernte Systeme zu konfigurieren oder auf Managementobjekte auf den Rechnern zuzugreifen und Informationen abzufragen.

2.3.5 IT-Dienstmodellierung Top-Down

Wir haben seither mit ITIL einen Bottom-Up Ansatz zum IT-Management und zur Erstellung eines IT-Dienstmodells betrachtet. Eine andere Art der Modellierung ist ein Top-Down Ansatz, bei dem ausgehend von den Geschäftsprozessen ein Modell für einen IT-Dienst erstellt wird.

ARIS/Modell Thiele

Thiele [Thiele05] beschreibt einen Ansatz, wie er momentan bei einigen Fachabteilungen von BMW im E-Bereich konzipiert wird. Dabei wird von einem Top-Down Ansatz ausgegangen, in dem ein IT-Dienst und die gesamte Infrastruktur nicht nur Top-Down modelliert sondern auch geplant wird. Dazu gehört unter anderem auch die Planung von Softwarearchitekturen und der IT-Infrastruktur.

Der von Thiele beschriebene Ansatz orientiert sich sehr stark an ARIS und verwendet ARIS auch zur Modellierung.

Abbildung 21 – IT-Management Top-Down

Überspitzt lässt sich ein Top-Down Ansatz wie in Abbildung 21 darstellen. Die fehlende Beschreibung der IT-Infrastruktur ist bei einigen dieser Ansätze charakteristisch. Tatsächlich finden sich im Ansatz von Thiele keine Schemata zur Modellierung der IT-Infrastruktur, welche nicht im Fokus der Arbeit lag. Im Gegensatz zu diesem überspitzten Vorgehen beinhaltet die Arbeit auch einen Bottom-Up Prozess zur Erfassung des Ist-Zustands. Eine These der Arbeit ist jedoch, dass dies nur zu Erfassung notwendig ist, da danach alle Veränderungen durch den Top-Down Ansatz automatisch immer zu einem konsistenten Modell führen.

Anwendungssysteme werden im Ansatz von Thiele nicht mit der IT-Infrastruktur verbunden, die Modellierung der IT-Infrastruktur erfolgt nur losgelöst von dieser. Alle Veränderungen in der Bebauung werden durch den Top-Down Prozess durchgeführt. Dies gilt zum Beispiel auch bei der Einführung von neuen Anwendungen. Die Anwendungen an sich werden nur als Einzelobjekte modelliert, eine Verteilung der Anwendungen auf verschiedene IT-Systeme wird nicht betrachtet.

Eine Integration mit den konkreten Tools zum IT-Management fehlt bei diesem Ansatz. Der Ansatz ist ein Spezialfall. ARIS beinhaltet mehr Informationen und erlaubt auch die Modellierung der IT-Infrastruktur. Dies erfolgt jedoch eher im Sinne eines Asset-Managements als eines Configuration-Managements.

Softwarekartographie

Auch die Softwarekartographie betrachte die IT-Dienstmodellierung von einer strategischen Ebene aus. Der Schwerpunkt liegt auf der Planung und der Weiterentwicklung von Bebauungsplänen. Man kann in diesem Zusammenhang eher von einer Planung und Steuerung im Großen sprechen, auch die Softwarekartographie verfolgt einen Top-Down Ansatz, der von den Geschäftsprozessen ausgeht und die IT-Infrastruktur nicht detailliert betrachtet. Der Fokus liegt hier deutlich stärker auf der Architektur.

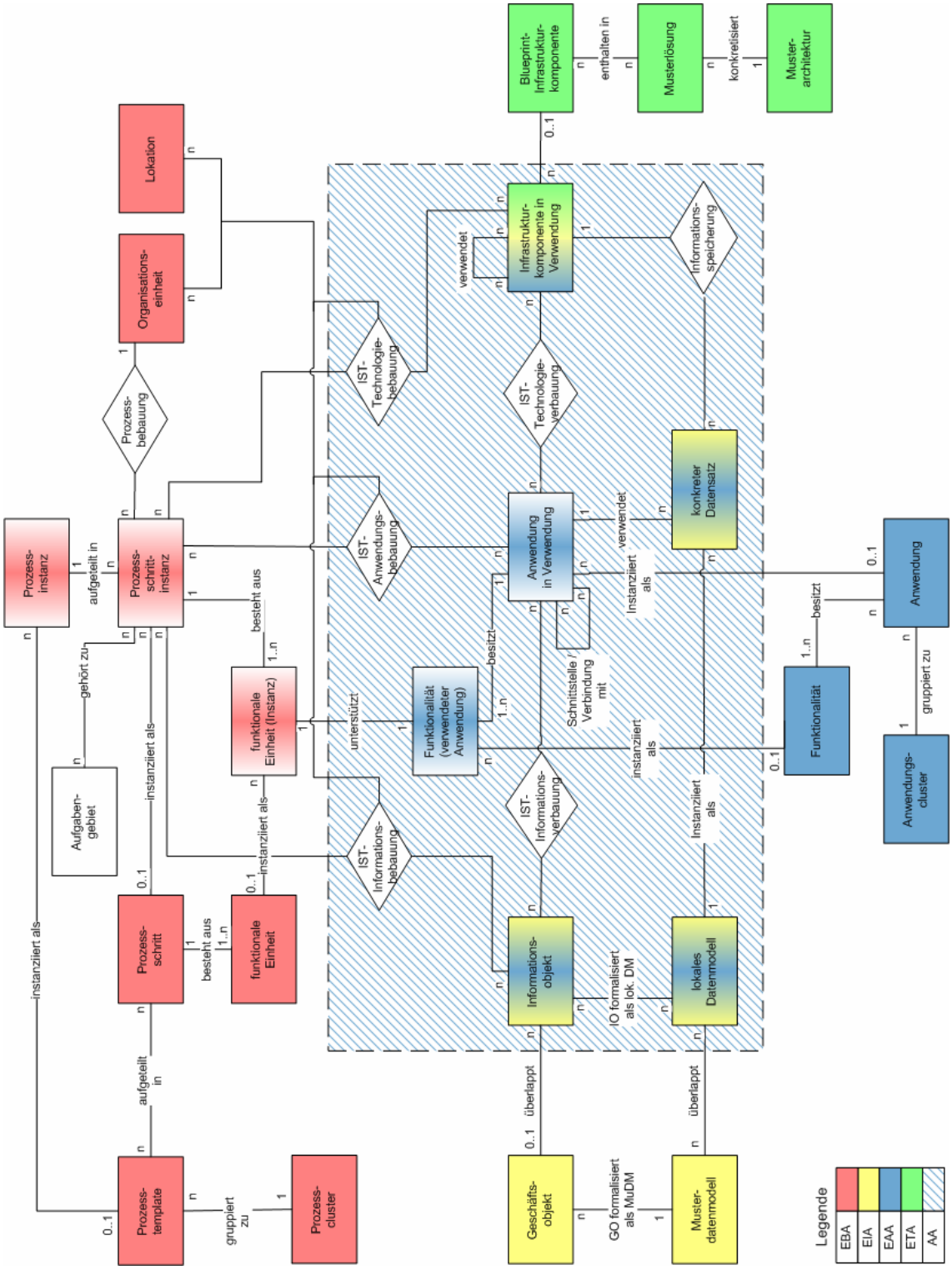

Abbildung 22 – Datenmodell zur IST-/PLAN-Bebauung nach Thiele (Quelle [Thiele05])

Darüber hinaus definiert die Softwarekartographie sehr genau die grafische Darstellung und den Inhalt von IT-Landkarten und Bebauungsplänen. Sie definiert vor allem auch Techniken zur Generierung von verschiedenen Perspektiven auf einen IT-Dienst und zur Transformationen der Karten.

Diese Transformationen können auch verwendet werden, um Informationen aus einer tieferen Ebene wie einem CIM Modell in einen Bebauungsplan zu integrieren. Die Softwarekartographie geht hier im Gegensatz zum Thiele Ansatz von verschiedenen anderen Modellen aus, die so integriert werden sollen.

2.4 Herausforderungen für die IT Heute

Wir haben bisher die aktuellen Rahmenbedingungen und Techniken für das Management von IT-Diensten beschrieben. Wir wollen in diesem Abschnitt auf die aktuellen Herausforderungen eingeben, die sich hieraus ergeben. Darüber hinaus stellen wir aktuelle Entwicklungen im Bereich der IT-Infrastruktur und der Softwarearchitektur vor und gehen auch hier auf die Auswirkungen auf das Management von IT-Diensten ein.

2.4.1 Herausforderungen aus der Unternehmensstruktur

Wir haben bereits beschrieben, dass die IT-Abteilung eines Unternehmens eingebettet in die Managementhierarchie ist. Sie stellt als Dienstleister Funktionen für andere Abteilungen zur Verfügung. Aber auch die IT-Abteilung ist den veränderten Rahmenbedingungen in einem Unternehmen unterworfen, die sich aus Änderungen in der Unternehmensstruktur ergeben.

Standardisierung, Outsourcing und die IT-Infrastruktur

Die Neuausrichtung von Unternehmen und großen Konzernen hat dazu geführt, dass alle Unternehmensteile auf ihren Beitrag zum Unternehmensziel hin überprüft und in Profit-Centern organisiert werden. Dies führt dazu, dass auch die IT-Abteilung heute ihren Wertbeitrag belegen muss und Investition vor allem auf eine Verringerung der Total Cost of Ownership (TCO[21]) und eine Verbesserung des Return on Investment (ROI) ausgerichtet sind. Dabei steht bei den meisten großen Konzernen eine Fokussierung auf die Kernkompetenzen im Mittelpunkt der Reorganisation.

Verschärft wurde die Diskussion im Bereich der IT durch einen sehr kontrovers diskutierten Artikel in der Harvard Business Review von Nicolas Carr mit dem Titel „IT Doesn't Matter". Die Grundthese des Artikels lautet, dass die IT aus sich selbst heraus nicht mehr dazu geeignet ist, einem Unternehmen einen Wettbewerbsvorteil zu verschaffen. Dies hat vor allem die Diskussion über das Outsourcing von IT-Dienstleistungen erneut angefacht. Richtig ist, dass es heute Aufgaben gibt, die so weit standardisiert sind, dass eine Auslagerung sinnvoll ist und zur Kosteneinsparung beiträgt. Ein Beispiel aus einer speziellen Branche dafür sind Geldautomaten, die von Unternehmen für verschiedene Banken betrieben werden.

Richtig ist aber auch, dass viele Kernprozesse eines Unternehmens heute untrennbar mit den IT-Systemen verbunden sind. Diese Kernprozesse tragen dabei sehr wohl zu einem Wettbewerbsvorteil des Unternehmens bei. Die zentrale Frage in diesem Zusammenhang ist also, ob und wie sehr einzelne IT-Prozesse in einem Unternehmen und der jeweiligen Branche standardisiert sind. Darüber hinaus muss betrachtet werden, wie diese IT-Dienste beim Wettbewerber erbracht werden und ob nicht gerade die Art und Weise der Erbringung im eigenen Unternehmen einen Vorteil gegenüber anderen darstellt. Erst danach lässt sich die Frage beantworten, ob die IT in diesem Fall einen Beitrag zum Wettbewerbsvorteil liefert, oder aber die Prozesse im Sinne der Nutzung von Economies of Scale besser an einen externen Dienstleister vergeben werden, der denselben Dienst in selber Güte für unterschiedliche Kunden anbietet.

Um die Frage aus der Perspektive der IT-Infrastruktur beantworten zu können, ist detailliertes Wissen über die Beziehungen der Geschäftsprozesse zur IT-Infrastruktur unerlässlich. Wichtig ist in diesem Zusammenhang auch, dass selbst bei standardisierten Prozessen die eigene IT immer noch einen Wettbewerbsvorteil darstellen kann, wenn sie sich durch sehr niedrige Kosten, eine große Flexibilität oder spezielle Features auszeichnet. Und dass in diesem Fall auch

[21] TCO, Total Cost of Ownership. Neben den Kosten für die Anschaffung fließen hier vor allem auch die Kosten für den Betrieb eines IT-Systems ein. Diese können die Anschaffungskosten deutlich übersteigen.

Investitionen in die IT dieser Prozesse zu einem Wettbewerbsvorteil führen können. Wichtig ist in diesem Zusammenhang auch, ob sich der Dienst überhaupt aus der Gesamtstruktur des Unternehmens herauslösen lässt, oder ob die Verflechtungen zu anderen Diensten so stark sind, dass eine Herauslösung dieses Prozesses gar nicht möglich ist. Diese Betrachtung muss dabei nicht nur auf der Ebene der Geschäftsprozesse, sondern auch auf der Ebene der Software- und der IT-Infrastruktur erfolgen.

Rechtliche Rahmenbedingungen

Wir haben bereits angedeutet, dass auch rechtliche Rahmenbedingungen immer mehr Einfluss auf das Management von IT-Diensten nehmen. Wir werden im Folgenden beispielhaft rechtliche Vorgaben vorstellen, dies entspricht keiner vollständigen Aufstellung. Es verdeutlicht aber vor allem, dass sich Vorgaben aus verschiedenen Richtungen ergeben können, die für ein Unternehmen bindend sind und es sich dabei nicht immer nur um gesetzliche Vorgaben handeln muss.

SOX

Der Sarbanes-Oxley-Act regelt in den USA Vorschriften für alle inländischen und ausländischen Kapitalgesellschaften, die an amerikanischen Börsen wie der New York Stock Exchange gelistet sind. Entstanden nach den Skandalen rund um Enron und Worldcom beschreibt SOX Vorschriften zur Veröffentlichung von Unternehmenszahlen, sowie die Art und Weise wie diese ermittelt werden.

Auf den ersten Blick hat dies keinen Einfluss auf die IT. SOX regelt aber auch, dass das Unternehmen, im speziellen die Unternehmensführung vertreten durch den CEO und den CFO, für die Prozesse zur Ermittlung der Finanzdaten haftet. Diese Informationen werden heute meist von IT-Systemen ermittelt, die entsprechenden Prozesse von IT-Systemen gesteuert. Deswegen ist der CIO eines Unternehmens in diesem Fall mitverantwortlich für die korrekte Ermittlung der Finanzinformationen.

Darüber hinaus sieht SOX auch einen Prüfungs/Audit-Prozess für diese Verfahren vor, bei dem die Ermittlungsmethoden und Prozesse des Unternehmens überprüft und von einem externen Gutachter als angemessen bewertet werden müssen. Die Wikipedia beschreibt folgende Auswirkungen von SOX auf die IT eines Unternehmens:

- Understanding the organization’s internal control program and its financial reporting process.
- Mapping the IT systems that support internal control and the financial reporting process to the financial statements.
- Identifying risks related to these IT systems.
- Designing and implementing controls designed to mitigate the identified risks and monitoring them for continued effectiveness.
- Documenting and testing IT controls.
- Ensuring that IT controls are updated and changed, as necessary, to correspond with changes in internal control or financial reporting processes.
- Monitoring IT controls for effective operation over time.
- Participation by IT in the Sarbanes-Oxley project management office.

Quelle: [WP-ITCont]

Beispiele für Anwendungssysteme, die von SOX betroffen sein können sind Datenbanken, Transaktionsmonitore, Workflow- und ERP-Systeme.

An die IT-Prozesse eines Unternehmens werden, wenn das Unternehmen unter die Rahmenbedingungen von SOX fällt, hohe Anforderungen gestellt, die entsprechend auch von Wirtschaftsprüfern oder anderen Prüfgesellschaften überwacht und zertifiziert werden. Dies gilt auch für die IT-Infrastruktur und deren Absicherung im Bezug auf Ausfallsicherheit und Datenschutz.

Andere Anforderungen mit Auswirkungen auf die IT sind untern anderem:

- Section 404

  Section 404 regelt das interne Kontrollsystem für den Vorgang der Ermittlung von Financial Reporting Daten. Damit muss sichergestellt werden, wie die Finanzdaten ermittelt und verwaltet werden und welche Risiken bei diesem Prozess vorhanden sind, besonders in Bezug auf die dabei verwendeten Systeme.

- Section 409

  Section 409 regelt, dass alle Informationen über Veränderungen in der Finanzstruktur eines Unternehmens in Echtzeit zur Verfügung stehen müssen. Dies soll dem Zurückhalten von Informationen vorbeugen. Daraus ergeben sich aber auch Anforderungen an die Arbeitsweise und Architektur der eingesetzten IT-Systeme.

- Section 802

  Section 802 regelt strafrechtliche Konsequenzen bei der Veränderung von Dokumenten. Um diese Veränderung feststellen zu können, müssen die Informationen auf der Ebene der IT-Systeme entsprechend geschützt sein.

SOX richtet sich eigentlich an die Corporate Governance Ebene und regelt eigentlich nur Bilanzierungsvorschriften. Durch den hohen Einfluss der IT-Dienste im Unternehmen haben diese Regelungen jedoch auch Auswirkungen auf die IT-Infrastruktur.

BASEL II

Basel II umfasst die Eigenkapitalvorschriften, die vom Basler Ausschuss für die Bankenaufsicht vorgeschlagen wurde. Basel II soll damit die Kreditvergabe in Europa reformieren. Neben den Vorschriften für Banken, die auch auf deren Prozesse Einfluss haben und damit Änderungen an den IT-Diensten verlangen, beinhaltet Basel II auch eine Komponente, die für alle Unternehmen relevant ist.

Basel II verlangt von den Banken, Kredite an Kunden relativ zum Ausfallrisiko, mit Eigenkapital zu hinterlegen. Das Risiko berechnet sich dabei nicht nur aus der Branche, in der der Kreditnehmer tätig ist, sondern auch nach dem Kreditnehmer an sich. Dazu wird der Kreditnehmer einem Rating unterzogen und das Risiko eines Kreditausfalls durch die Bank nach Vorgaben, die von Basel II vorgegeben sind[22], bewertet.

Daraus ergeben sich unter anderem Forderungen der Banken an die Unternehmen, ein Risikomanagement im Unternehmen zu etablieren. Für ein Unternehmen kann sich durch die Etablierung und vor allem die Verbesserung des Risikomanagements eine Verbesserung der Kreditbedingungen ergeben. Je stärker ein Unternehmen von seiner IT-Infrastruktur abhängt, desto wichtiger ist auch hier das Risikomanagement.

BASEL II ist ein weiteres Beispiel, wie in diesem Fall Vorgaben von Banken zur Kreditvergabe, also keine gesetzlichen Vorgaben, Einfluss auf das IT-Management haben. Im Gegensatz zu SOX, das vor allem für große Aktiengesellschaften relevant ist, betrifft BASLE II auch kleine und mittlere Unternehmen, die seither aus betrieblichen Gründen eventuell noch kein formales IT-Governance etabliert haben.

HIPAA

Der Health Insurance Portability and Accountability Act (HIPAA) ist ein Beispiel für rechtliche Rahmenbedingungen, die in einer speziellen Branche gelten. Er wurde 1996 vom US-Kongress erlassen und soll unter anderem Verbesserungen im US-Gesundheitswesen bewirken.

[22] In Zusammenarbeit mit der Bankenaufsicht BaFIN ist ein abweichen von den Standards möglich, wenn die BaFIN das Verfahren zur Risikobewertung der Bank als äquivalent zu den Basel II Vorgaben bewertet.

Ein wichtiger Teil der HIPAA Vorschriften betrifft aber auch die IT im Gesundheitswesen. HIPAA regelt mit den Vorgaben HIPAA/EDI die Datenformate zum Austausch von Informationen im Gesundheitswesen. Die Kopplung von Systemen verschiedener Anbieter wird damit vereinfacht und in einheitliches Datenschema vorgegeben. Werden Gesundheitsdaten ausgetauscht, ist dieses Datenformat zu verwenden.

Die Sicherheitsregeln stellen darüber hinaus die Vertraulichkeit der Daten sicher und stellen Anforderungen an die Prozesse zum Umgang mit den Daten. Diese Prozesse müssen zertifiziert sein. Unter anderem muss genau festgelegt werden, welche Mitarbeiter Zugriff auf die Daten erhalten und wie der Zugriff auf die Daten geregelt ist. Gleiches gilt für die Datensicherung und die Datensicherheit.

Die IT-Governance Vorgaben werde in diesem Fall sehr stark durch HIPAA vorgegeben und müssen in einem betroffenen Unternehmen entsprechend umgesetzt werden.

Zusammenfassung

Es gibt mittlerweile sehr viele verschiedene rechtliche Regelungen, die Einfluss auf die IT und IT-Governance haben. Die Regelungen beziehen sich dabei, wie dargestellt, teilweise auf die Bilanzierung und das Risikomanagement im Unternehmen, teilweise aber auch auf sehr spezielle technische Aspekte wie HIPAA. Gerade bei Änderungen an diesen Vorschriften ist ein detailliertes Wissen um die Konsequenzen für das Unternehmen, als auch die bereits durchgeführten Arbeiten notwendig, um den Regelungen zu entsprechen und dies auch darstellen zu können.

Mit CobiT steht ein Framework zur Verfügung, das Regeln zum IT-Governance festlegt und für das entsprechend Abbildungen auf SOX und ITIL vorliegen. Damit ist die Einhaltung der entsprechenden rechtlichen Rahmenbedingungen einfacher möglich. Darüber hinaus trägt dies zu einer Standardisierung der IT bei. Wichtig ist jedoch, dass die entsprechenden Prozesse und Rahmenwerke an das Unternehmen angepasst und im Unternehmen auch gelebt werden.

In Zukunft werden regulatorische Einschränkungen immer mehr zunehmen und einen immer größeren Einfluss auf die IT-Infrastruktur bekommen. Dadurch wird das IT-Management komplexer und die Vorgaben zur Dokumentation und Steuerung vielschichtiger. Gerade das Beispiel BASEL II zeigt, dass davon auch kleine und mittlere Unternehmen in Zukunft in einem deutlich größeren Maß betroffen sein werden.

2.4.2 Technische Herauforderungen und neue Technologien

Neben den bereits beschriebenen rechtlichen- und organisatorischen Rahmenbedingungen und Herausforderungen an die IT ergeben sich heute auch technische Herausforderungen, die teilweise von unternehmerischen Vorgaben ausgehen oder auf technische Neuerungen und Verbesserungen zurückzuführen sind, auf die wir im folgendem eingehen.

Eine wichtige Veränderung ergibt sich aus der Art und Weise, wie IT-Systeme heute eingesetzt werden. Moderne Kommunikationsmittel wie Handys und PDAs ermöglichen dem oft auch als Information Worker bezeichneten Mitarbeiter im Unternehmen den Zugriff auf Informationen an jedem Ort. Neben der Anpassung der Daten auf die unterschiedlichen Geräte, erfordert dies auch ein angepasstes Sicherheitskonzept. Getrieben wird diese Entwicklung auch dadurch, dass Benutzer die Bereitstellung bestimmter Dienste (Email-Push, VPN) einfordern und die IT-Abteilung auf diese Anforderungen reagieren muss.

Ein Beispiel ist hier der Blackberry Dienst der Firma Research in Motion (RIM). Mit dem Dienst werden Emails, die auf einem Microsoft Exchange oder IBM Lotus Notes Server eintreffen, direkt zu RIM in eines der drei Rechenzentren übertragen. Von dort aus werden die Daten dann an ein mobiles Gerät, oft ebenfalls als Blackberry bezeichnet, oder das Handy des Empfängers als Email über das jeweilige Mobilfunknetz weitergeleitet. Anwender müssen so nicht mehr aktiv überprüfen, ob sie eine Nachricht erhalten haben, sondern bekommen diese beim Eintreffen angezeigt. Darüber hinaus können mit dem Blackberry auch Emails erstellt und über das Mobilfunknetz versendet werden. Der Dienst ist gerade bei Mitarbeitern, die oft Unterwegs sind, sehr beliebt.

Eine Entscheidung über die Einführung einer neuen Technologie wird oft ohne eine genaue Analyse der Auswirkungen getroffen. Eine öffentliche Diskussion über die Tatsache, dass das RIM Gateway direkt in die Exchange/Notes Infrastruktur integriert wird und alle Emails, die innerhalb der Unternehmensinfrastruktur versendet werden, abgreifen und weiterleiten kann, hat zu Bedenken im Bezug auf die Datensicherheit geführt. RIM wäre mit dieser Technik in der Lage, alle Maßnahmen zur Datensicherheit in einem Unternehmen zu unterlaufen und die Technik zur Betriebsspionage einzusetzen. Darüber hinaus werden so Daten eventuell in Länder übertragen und gespeichert, in denen andere Datenschutzrichtlinien gelten. Damit würden die hohen Datenschutzrichtlinien, die in Europa gelten und von den Unternehmen umzusetzen sind, ausgehöhlt. Die Tatsache, dass zwei der drei RIM Rechenzentren nicht in Europa liegen und die Datenschutzrichtlinien einen Transfer in Länder mit einem niedrigeren Schutzstandard nicht erlauben, stellt hier ein mögliches Problem dar. Darüber hinaus hat eine Diskussion der Auswirkungen der Platzierung des RIM Gateways in der Exchange Infrastruktur und deren Ausführungsrechte dazu geführt, dass die Audi AG [AUDIRIM] im Juni 2006 kurz davor war, ein Nutzungsverbot für die von der Unternehmens-IT ausgegebenen Blackberrys zu erteilen. In diesem Beispiel wurden offenbar bei der Einführung eines Dienstes in die IT des Unternehmens nicht alle Auswirkungen und die Details der Implementierung auf die Infrastruktur beachtet und entsprechend bewertet. Dies verdeutlicht, dass entsprechende Entscheidungen über IT-Dienste nicht nur auf einer abstrakten Ebene und Sicht auf die Dienste getroffen werden können, sondern Entscheidungen alle Ebenen betrachten müssen, um alle Auswirkungen der Einführung zu betrachten.

Wir werden im Weiteren andere Beispiele für technische Entwicklungen und deren Konsequenzen auf die Bereitstellung von IT-Diensten beschreiben.

SOA - Service-orientierte Architekturen

Service-orientierte Architekturen (SOA) werden heute oft als eine neue Art der Softwarestrukturierung angesehen. Wichtig ist jedoch, dass die Serviceorientierung nicht nur auf die Programmierung wirkt. Dienste und Prozesse müssen entsprechend modelliert und beschrieben werden. Service-orientierte Architekturen bieten entscheidende Vorteile, auch wenn einige der Konzepte, auf denen Sie basieren, teilweise schon lange bekannt sind. Vorteile von SOA sind:

- Zugriff für unterschiedlichste Endgeräte

  Wie bereits beschrieben existieren heute eine Vielzahl von Clients. Während Mainframesysteme fast ausschließlich von Terminals aus bedient wurden, haben die Anwender heute eine breite Palette von Clients zur Verfügung, von Handhelds und PDAs bis zu Notebooks und TabletPCs. Alle diese Clients unterscheiden sich in ihrer Benutzeroberfläche, der Art der Benutzereingabe und der Rechenleistung. Eine SOA kann dazu beitragen, Services unabhängig vom Endgerät zur Verfügung stellen zu können, indem die Adaption der Darstellung von den fachlichen Services getrennt wird.

- Keine feste Programmierung von Prozessen

  Wie bereits beschrieben bieten Systeme wie Biztalk die Möglichkeit, Workflows nicht nur in Programmcode zu hinterlegen, sondern diesen auch deklarativ in BPEL zu beschreiben. Damit wird es möglich, Applikationen wesentlich einfacher zu ändern und an Veränderungen anzupassen. Geschäftsprozesse werden nicht mehr in Software hinterlegt, sondern direkt modelliert und ausgeführt. Dabei steht die fachliche Modellierung und Beschreibung im Fordergrund, die Sprache orientiert sich nicht an einer klassischen Programmiersprache sondern an den Geschäftsprozessen.

- Integration von Webservices in Standardapplikationen

  Ein Beispiel ist Microsoft Office. Alle Officeapplikationen sind in der Lage, Daten aus Webservices zu konsumieren. Damit wird es dem Anwender möglich, direkt mit dem Datacenter zu interagieren und neue „Applikationen“ zu schaffen, die nicht von einer zentralen IT entwickelt oder bereitgestellt werden. Damit wird der Benutzer zum Softwarearchitekten und Softwareentwickler, der aus Basisdiensten neue Applikationen schafft. Dies erfordert ein entsprechendes Management der Dienste, da eine zentrale

Planung und Entwicklung der neuen Dienste hier nicht stattfindet. Ein Beispiel hierfür ist die Integration von SAP Netweaver mit Duet in Office.

Serviceorientierte Architekturen bieten so nicht nur neue Möglichkeiten, Applikationen schneller und flexibler zu gestalten, sie stellen auch neue Anforderungen an das Management entsprechender Infrastrukturen, vor allem an das Availability Management und an das Release Management, da die Nutzung der Basisdienste nicht mehr zentral gesteuert und geplant werden kann. Ob ein Service genutzt wird oder nicht, kann nicht mehr nur durch eine Betrachtung aller zentral betriebenen Applikationen entschieden werden. Eventuell müssen hier Zugriffsstatistiken auf den Dienst betrachtet werden. Dienste wie UDDI versuchen hier, neben der Bereitstellung eines Verzeichnisses für unterschiedliche Dienste auch unterschiedliche Versionen eines Dienstes darzustellen. Die Pflege der verschiedenen Versionen macht dabei die Bereitstellung komplexer, da die Zahl der Services mit jeder neuen Version zunimmt.

Virtualisierte Infrastrukturen

Wir haben als eine wichtige unternehmerische Anforderung an die IT die Verbesserung des Return on Investment (ROI) beschrieben. Eine Möglichkeit, den ROI zu verbessern ist, die Auslastung der Serversysteme zu erhöhen. Die typische Auslastung heutiger Server liegt nach Studien unterschiedlicher Analysten zwischen 5%-20%. Ähnliche Zahlen ergaben sich bereits vor Jahren im Bereich Direct attached Storage, also bei Speichersubsystemen, die direkt an einzelne Server angeschlossen sind. Hier konnten durch Virtualisierungslösungen und die Zusammenfassung der Speichersysteme zu einem Storage Area Network (SAN) Steigerungen auf durchschnittlich 70-80% erreicht werden, indem die Ressourcen durch mehrere Server genutzt wurden. Dadurch können freie Kapazitäten Servern mit höherem Speicherbedarf zugeordnet werden. Darüber hinaus ist durch die Abstraktion in einem SAN die Verlagerung einzelner Dienste von einem Server zu einem anderen deutlich einfacher möglich, da die Datenbereiche nur durch eine Änderung der Zuordnung von einem Server zu einem anderen verlagert werden können[23].

Virtualisierte Infrastrukturen können hier einen deutlichen Beitrag leisten, diese Konzepte aus dem SAN Bereich auch auf Serversysteme zu übertragen und hier entsprechende Einsparungseffekte zu erzielen. Erste Erfahrungen in diesem Bereich zeigen, dass diese Ziele erreicht werden können. Zu beachten ist dabei aber die Steigerung der Komplexität der Systeme und die höheren Anforderungen an die Verfügbarkeit der physikalischen Systeme, da ein Ausfall des physikalischen Systems mehrere virtuelle Systeme betrifft.

Zu unterscheiden sind die verschiedenen Arten der Virtualisierung, die jeweils unterschiedliche Konsequenzen für die Erbringung von IT-Diensten haben:

- Anwendungsvirtualisierung: Instanzen auf der Ebene von Applikationen

  Die einfachste Möglichkeit, virtuelle Dienste zu Verfügung zu stellen, bieten heute Applikationen wie Datenbanken und Webserver. Auf einem Server werden durch die entsprechenden Applikationen mehrere Instanzen eines Dienstes ausgeführt. Die Trennung der verschiedenen Instanzen hängt dabei von der jeweiligen Anwendung ab, bei einem Microsoft SQL Server haben die verschiedenen Datenbanken verschiedene Namen und werden auf unterschiedlichen TCP/IP Ports ausgeführt. Ein Spielart sind hier Clustersysteme wie Microsoft Cluster Services, die neben verschiedenen Instanzen auch Hochverfügbarkeit ermöglichen.

  Mit Blick auf den Server ist dies noch keine wirkliche Virtualisierung. So können aber viele gleiche Serverdienste auf einen großen Server konsolidiert werden, man spricht in diesem Zusammenhang auch von Scale-in. Dies steigert die Auslastung des Servers, ermöglicht durch die Migration auf einen Cluster auch die Steigerung der Verfügbarkeit bei einem Ausfall oder Wartungsarbeiten und trägt zur Vereinheitlichung der

[23] Darüber hinaus bringt ein SAN Geschwindigkeitsvorteile, da multiple Pfade zwischen Server und Storage die Geschwindigkeit steigern können und in einem SAN Storage-Subsystem meist mehrere Festplatten als in einem Direct attached Storage DAS System zur Verfügung stehen

Diensterbringung bei. Auf der anderen Seite generiert dies aber eine komplexere Struktur und schafft zusätzliche Abhängigkeiten zwischen Systemen. Auch die Auswirkungen eines Ausfalls betreffen in diesem Szenario mehrere Dienste.

- Servicevirtualisierung: Flexframe, ASCC, SUN Gridcontainer für SAP

  Produkte wie Flexframe tragen das Konzept der Anwendungsvirtualisierung weiter und ermöglichen es, komplexe Systeme wie SAP Netweaver mit all seinen Komponenten (Webserver, Datenbanken) dynamisch zur Verfügung zu stellen. Die Managementsysteme sind in der Lage, die Verteilung der Software auf verschiedene Server zu beeinflussen. Ändern sich die Anforderungen an einen IT-Dienst, oder fällt ein System aus, werden die Komponenten einer entsprechenden Anwendung neu verteilt und die notwendigen Instanzen auf den entsprechenden IT-Systemen gestartet.

  Dadurch ergibt sich für Systeme wie SAP Netweaver eine Optimierung der Laufzeitumgebung, die deutlich flexibler als eine statische Installation ist. Die Managementsysteme beinhalten ein Modell der Anwendung und können diese verwalten. Sie sind aber auf die entsprechenden Anwendungen beschränkt. Entsprechende Systeme können einen Beitrag zur Optimierung der IT-Infrastruktur leisten, vor allem weil damit eine Umverteilung von Anwendungen je nach Auslastung einzelner Dienste sehr einfach möglich ist. Durch die Auflösung der festen Verteilung müssen die Informationen über die aktuelle Verteilung aus dem Managementsystem entnommen werden. Die Migration auf neue Versionen des Anwendungssystems erfordert auch eine Anpassung des entsprechenden Managementsystems.

- Softwarevirtualisierung: Virtual PC/Server, VMWare Workstation/GSX Server

  Die Lösungen zur Softwarevirtualisierung stellen virtuelle Maschinen zur Verfügung, in die ein Betriebsystem installiert wird. Diese Lösungen emulieren in dieser Umgebung die Hardware eines Servers und stellen für das Gastbetriebsystem eine Laufzeitumgebung dar, die sich nicht von einem physikalischen Rechner unterscheidet. Die jeweiligen Serverprodukte bieten darüber hinaus die Möglichkeit zum Start mehrer Instanzen, die Unterstützung für multiple Netzwerkkarten, etc. Interessant sind hier auch Konzepte zur Hochverfügbarkeit, bei denen eine Entkopplung zwischen den Hostsystemen und den virtuellen Maschinen erreicht werden kann. Die virtuellen Maschinen lassen sich bei diesen Lösungen im Betrieb von einem Hostrechner zu einem anderen verlagern. Dazu wird der Zustand der virtuellen Maschine auf eine gemeinsame Festplatte gespeichert, die Maschinenkonfiguration auf einen anderen Server verlagert und dort der gesicherte Zustand wieder in den Speicher des Rechners geladen.

  Diese Lösungen ermöglichen die deutliche Steigerung der Auslastung von physikalischen Servern, da viele Server im Sinne eines Scale-In auf einen Server zusammengefasst werden können. Die meisten Produkte bieten hier auch Lösungen an, mit denen sich physikalische Maschinen in virtuelle Maschinen verlagern lassen.

  Wie schon bei den Anwendungssystemen beschrieben führt dies jedoch dazu, dass bei einem Ausfall des physikalischen Servers deutlich mehr Systeme vom Ausfall betroffen sind. Die beschriebenen Hochverfügbarkeitslösungen können die Auswirkungen hier mildern, bei einem Ausfall eines physikalischen Knoten ist aber eine Ausfallzeit nicht zu vermeiden, da die virtuellen Instanzen erst wieder neu gestartet werden müssen.

- Hypervisor: XEN, Windows 2008 Server, VMWare ESX/V-Motion

  Hypervisor-basierte Systeme wie XEN [XEN03] oder der kommende Hypervisor im Windows 2008 Server funktionieren ähnlich wie Softwarevirtualisierung. Sie basieren jedoch nicht auf einem Standardbetriebsystem als Hostbetriebsystem, sondern verwenden dafür spezielle Betriebsysteme. Diese zeichnen sich dadurch aus, dass sie nur grundlegende Betriebsystemfunktionen bieten. Die Verkleinerung des Hostbetriebsystems bewirkt vor allem, dass weniger Systemressourcen durch den Host gebunden werden. Auch wird dadurch die Anzahl der potentiellen Neustarts wegen Software-

/Sicherheitsupdates verringert, da deutlich weniger Komponenten im Hostbetriebsystem vorhanden sind.

Eine Spezialität von Systemen wie XEN ist die so genannte Paravirtualisierung. Dabei wird das Gastbetriebsystem angepasst und „lernt“, dass es in einer virtuellen Umgebung ausgeführt wird. Dadurch können spezielle Aufgaben, die in einer emulierten Umgebung sehr aufwendig sind, direkt an den Hypervisor weitergegeben werden, der diese auf der tatsächlichen Hardware deutlich schneller durchführen kann.

Alle Einschränkungen für Softwarevirtualisierung gelten auch für Hypervisorsysteme. Auch hier entstehen weitere Abhängigkeiten zwischen den Systemen und der Ausfall eines physikalischen Hosts hat deutlich größere Auswirkungen auf die Gesamtinfrastruktur, da mehrere virtuelle Server betroffen sind.

- Hardwarevirtualisierung: Mainframes, IBM XA, AMD Pacifica, Intel VT/VT-d

  Lösungen zur Hardwarevirtualisierung setzen auf einem noch tieferen Niveau an als Hypervisorsysteme. Sie werden realisiert durch die Integration der Virtualisierungsfunktionen in die Prozessoren und das BIOS des Servers. Sie sind damit den Hypervisorsystemen sehr ähnlich, benutzen diese teilweise zur Steuerung der Virtualisierung.

  Mainframesysteme stellen entsprechende Funktionen schon seit langem zur Verfügung. Mit AMD Pacifica und Intel VT stehen diese Funktionen nun auch x86/64 basierten Systemen zur Verfügung. Diese Funktionen können durch Hypervisorssteme wie XEN dann genutzt werden, um effektiver mit den Ressourcen des Systems umzugehen und weniger Komponenten per Software virtualisieren zu müssen. Damit kann XEN auf Anpassung des Betriebsystems verzichten und dem Gastbetriebsystem teilweise die Kontrolle über die physikalische Hardware übergeben. Eine Anpassung des Gastbetriebsystems muss dann nicht erfolgen.

  Neben der Steigerung der Auslastung bieten neue Techniken wie Intel VT-d vor allem auch eine Verbesserung für die Sicherheit virtuelle Maschinen, indem auch die Ein-/Ausgabekanäle virtualisiert werden und so eine Abschottung der virtuellen Maschinen voneinander auch bei der Kommunikation erreicht wird. Darüber hinaus zeichnen sich hier Techniken zur Absicherung von virtuellen Maschinen mit Trusted Computing Techniken ab. Dies ist notwendig, um nur zertifizierte virtuelle Maschinen auf einem Rechner ausführen zu können. Es existieren mittlerweile mehrere Proof-of-Concept Implementierungen von Rootkits, die die neuen Virtualisierungsmöglichkeiten nutzen, um sich vor Virenscannern und dem Betriebsystem zu verstecken. Dabei wird das gesamte Betriebsystem einfach in eine virtuelle Maschine verlagert, die vom Rootkit gesteuert wird. Das Betriebsystem hat keine Möglichkeit mehr, das Rootkit zu erkennen.

  Auch die Hardwarevirtualisierung hat zur Konsequenz, dass durch ein Scale-In mehrere physikalische Rechner auf einen Server verlagert werden, dadurch wird der Ausfall des Servers kritischer.

Allen Techniken zur Virtualisierung ist gemein, dass Sie zusätzliche Abhängigkeiten generieren. Durch die zusätzlichen Schichten und die teilweise transparente Verteilung der virtuellen Maschinen wird die Bewertung und Analyse von Ausfällen deutlich schwieriger. Darüber hinaus entstehen so neue Abhängigkeiten der virtuellen Maschinen von physikalischen Servern und die Auswirkungen des Ausfalls eines physikalischen Servers werden dadurch deutlich gravierender. Gerade aber die notwendige Steigerung der Auslastung von Servern wird zum verstärkten Einsatz der Virtualisierung sorgen.

Ein weiterer wichtiger Aspekt ist, dass sich virtuelle Maschinen sehr leicht und ohne Änderung der Maschine auf leistungsfähigere Hardware verlagern lassen. Dadurch können entsprechende Maschinen über die Zeit sehr einfach wachsen und durch schnellere Hostsysteme beschleu-

nigt werden. Darüber hinaus ist ein Backup einer entsprechenden Maschine ganz einfach durch das Sichern der Datei im SAN möglich[24].

Neue Rechnerkonzepte

Wir haben bei den Virtualisierungskonzepten schon einige neue Techniken auf der Ebene der Hardware mit Intel-VT angedeutet. Ein aktueller und anhaltender Trend ist die Einführung von Multi-Core Prozessoren. Dabei werden zwei oder mehr Prozessorkerne auf ein Die gebracht. Hier entsteht dann bereits beim Einsatz eines Prozessors ein Mehrprozessorrechner. Im Mainstreambereich von Intel und AMD sind bis Ende des Jahres 4-Kern Prozessoren angekündigt, durch typische Mainboards, wie sie heute eingesetzt werden, lassen sich daraus sehr einfach Server bauen, die 16 Prozessorkerne beinhalten. SUN geht hier noch weiter und hat mit dem Niagara-2 einen Prozessor angekündigt, der acht Kerne beinhaltet und pro Kern acht Threads ausführen kann. Dies verdeutlicht, dass einzelne Rechnersysteme immer leistungsfähiger werden, auf der anderen Seite aber auch spezielle Anforderungen an die Software stellen, denn nur Software, die parallel Aufgaben abarbeiten kann, nutzt auch wirklich die Gesamtleistung eines entsprechenden Prozessors. Dies muss vor allem bei der Verteilung von Softwarekomponenten auf die Serversysteme von einem Administrator oder einem entsprechendem Managementsystem beachtet werden.

Außerdem ergibt sich bei den neuen Prozessoren eine weitere Änderung in der Architektur, die eine neue Herausforderung für die Software darstellt. Klassische PCs arbeiten mit einem Speicher, der dem gesamten Prozessor zur Verfügung steht. Man spricht hier auch von einem Uniform Memory Access (UMA) Konzept, da der Prozessor (oder mehrere Prozessorkerne/Prozessoren) über eine einzige Verbindung in gleicher Weise auf den gesamten Speicher zugreifen. Die Verbindung zwischen dem Prozessor (oder mehreren Prozessoren) erfolgt über die Northbridge zum Speicher, Intel bezeichnet diese auch als Memory Controller Hub. Die Abbildung 23 zeigt diese Architektur für ein Xeon System auf der rechten Seite. Die Bandbreite zum Speicher teilen sich alle Prozessoren. Dies führt aber bei vielen Prozessorkernen zu einem Problem, da das Speicherinterface zu einem Flaschenhals wird.

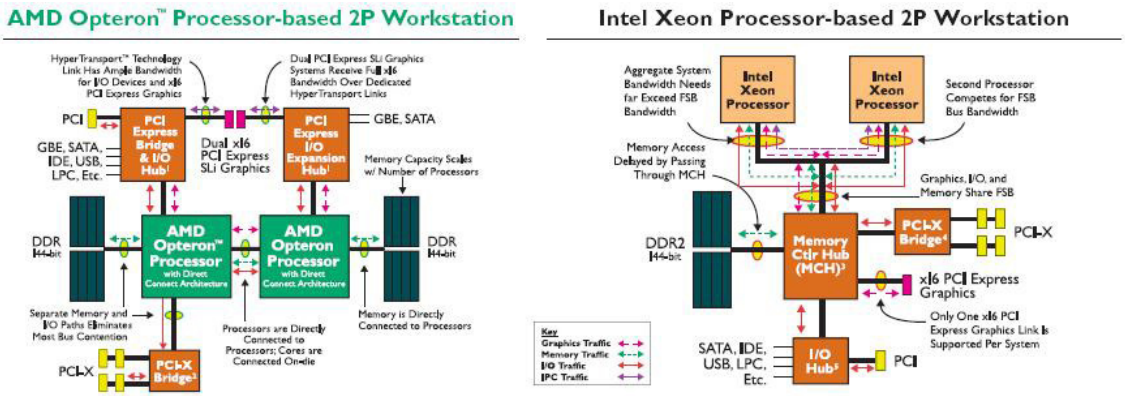

Abbildung 23 – NUMA und UMA Architektur (Quelle AMD)

Deswegen verwenden neue Prozessordesigns wie der AMD Opteron neue Speicherschnittstellen wie Hypertransport. Dabei werden die Prozessoren direkt mit einem Teil des Speichers verbunden, der dann dem Prozessor zugeordnet wird. Jeder Prozessor hat in dieser Architektur einen Teil des Speichers direkt angebunden. Die Prozessoren sind darüber hinaus untereinander mit Hypertransportkanälen verbunden. Muss ein Prozessor auf einen anderen Speicherbereich als den eigenen zugreifen, greift er über Hypertransport auf den Memory-Controller des

[24] Die Darstellung ist etwas vereinfacht, ein Backup im Betrieb erfordert, dass die Virtualisierungslösung dies unterstützt, wie zum Beispiel Microsoft Virtual Server 2005 R2 SP1 mit der Unterstützung von Volume Shadow Copies.

jeweiligen Prozessors zu, an den der Speicherbereich angebunden ist. Dies ergibt eine höhere Latenzzeit für den Speicherzugriff als bei einem lokalen Zugriff. Bei entsprechenden Rechnerarchitekturen spricht man deshalb auch von Non-Uniform Memory Access (NUMA) NUMA oder NUMA Architekturen. Die Managementsoftware, vor allem einer Virtualisierungslösung, muss entsprechende Eigenschaften der Architektur in die Planung der Verteilung der virtuellen Maschinen mit einbeziehen, um Engpässe durch die Rechnerarchitektur zu verhindern[25].

Neben diesen Veränderungen ergeben sich aber auch andere neue Rechnerkonzepte. FSC Bladeframe ist ein Beispiel für eine neue Rechnerarchitektur, die als nächste Stufe der Blades gesehen werden kann. Blades sind herkömmliche Server, die in einen Einschub gepackt sind und in einem Bladecenter über einen oder mehrere spezielle Steckerverbindungen betrieben werden. Komponenten zur Anbindung von Netzwerkverbindungen, Speicherlösungen und Stromversorgung werden zentral durch das Bladecenter zur Verfügung gestellt. Durch die Standardisierung der Blades wird das Management vereinfacht und eine höhere Packungsdichte erreicht. Bladeframe trägt diese Vorteile weiter und realisiert mit Blades eine Art Mainframe aus Standardkomponenten. Die Blades sind dabei reine Prozessorkonten mit lokalem Speicher, alle weiteren Funktionen werden in der Backplane implementiert. Aus den verschiedenen Prozessorknoten können dann durch Virtualisierungsfunktionen virtuelle Maschinen gebildet werden. Durch den Aufbau aus einfachen Basiskomponenten entsteht so ein sehr preiswertes, anpassbares System.

Die neuen Rechnerarchitekturen stellen erhöhte Anforderungen an das Betriebsystem zur optimalen Betriebsmittelverteilung, aber auch an Anwendungen wie Virtualisierungslösungen. Die wechselseitigen Abhängigkeiten müssen vor allem in Optimierungsstrategien mit einbezogen werden und machen die Planung der idealen Hardware für eine Softwarelösung noch schwieriger. Die Software und ihre Anforderung muss genau mit den Möglichkeiten der Hardware abgeglichen werden.

Sicherheit

Netzwerke und das Internet ermöglichen potentiell die Kommunikation alle angeschlossenen Systeme miteinander. Neue Sicherheitsarchitekturen versuchen heute, Sicherheitsfunktionen in diese Netze nachzurüsten. Oft fasst man diesen Begriff auch unter dem Schlagwort Trustworthy Computing oder Trusted Computing zusammen.

Wichtige Herausforderungen in diesem Bereich sind vor allem die Absicherung der IT-Systeme an sich. Benutzer sind mit Notebooks heute nicht mehr an einen physikalischen Ort gebunden, sie verbinden sich über unterschiedliche Netze mit dem Firmennetzwerk. Für das Management stellt dies entsprechende Herausforderungen an die Absicherung der Systeme, da eine Verbindung nicht mehr per se als sicher betrachtet werden kann. Konzepte wie Identity driven Networking und Windows 2008 Server Network Access Protection sorgen dafür, dass Rechner erst nach einer Überprüfung, ob sie den Sicherheitsrichtlinien des Unternehmens entsprechen, Zugriff auf die Daten des Firmennetzwerkes erhalten. Die Frage, ob ein System mit einem anderen kommunizieren kann ist damit nicht mehr nur auf die Existenz einer physikalischen Verbindung zurückführbar. In die Entscheidung muss der Zustand der jeweiligen Maschine mit einfließen. Firewallsysteme und Netzwerkkomponenten wie Switches setzen die Richtlinien für die Absicherung des Netzwerks durch und etablieren so logische Netzstrukturen, die deutlich komplexer als die physikalische Netzwerkstruktur sind.

Aber auch in den Rechnern selbst sind entsprechende Schutzmaßnahmen, wie im Abschnitt über Virtualisierung bereits beschrieben, notwendig. Gerade Notebooks gehen häufig verloren. Schlimmer als der finanzielle Verlust ist dabei meist der Verlust der Informationen an sich, die auf dem Gerät gespeichert wurden. Dies gilt umso mehr, wenn vertrauliche Informationen auf Handhelds gespeichert werden. Die Sicherung dieser Informationen wird in Zukunft noch mehr an Bedeutung gewinnen.

[25] Lösungen wie Virtual Server 2005 nehmen diese Optimierung hier vor, bzw. Warnen bei Konfigurationen, die die Leistungsfähigkeit der virtuellen Maschine einschränken

2.5 Fazit

Wir haben in diesem Kapitel das Szenario der Erbringung von IT-Diensten beschrieben. Die Vorgaben für das IT-Management ergeben sich nicht nur aus der Technik und deren Veränderung selbst, sondern sind integriert in die Managementhierarchie des Unternehmens.

Es existieren heute verschiedene Techniken zum Management und zum Umgang mit den Herausforderungen für das Management von IT-Diensten, an dem viele verschiedene Personen mit ihren jeweiligen Perspektiven beteiligt sind. Eine einheitliche Sicht auf diese Ebenen ist nicht vorhanden. Neue Technologien und technische Herausforderungen sorgen dafür, dass sich die IT-Infrastruktur laufend verändern und anpassen muss. Dabei ist es wichtig, wie am Beispiel von Blackberry beschrieben, dass Dienste und Funktionen nicht nur auf einer abstrakten und funktionalen Ebene betrachtet werden, sondern dass auch die Implementierungsdetails und die sich daraus ergebenden Änderungen und Konsequenzen in die Planung und Steuerung der IT-Infrastruktur mit einfließen. Dabei ist ein umfassendes Bild der vorhandenen Dienste notwendig, um Änderungen und deren Auswirkungen betrachten und beurteilen zu können.

Nach der Vorstellung der Rahmenbedingungen für den IT-Betrieb leiten wir im nächsten Abschnitt die Anforderungen für ein ganzheitliches Modell eines IT-Dienstes ab.

3 Anforderungen an eine Modellierung von IT-Diensten

Wir haben im vorigen Kapitel die Herausforderungen, die sich heute an Unternehmen und damit direkt auch an die IT-Abteilung ergeben, vorgestellt. Darüber hinaus haben wir exemplarisch die wichtigsten Techniken zum Management von IT-Diensten dargestellt. Festzuhalten bleibt, dass es heute kein umfassendes Managementwerkzeug oder eine Beschreibungstechnik gibt, die einen IT-Dienst sowohl auf der technischen, wie auch auf der fachlichen Ebene darstellten kann. Auch die Abhängigkeiten und Beziehungen sind heute nicht in einem Modell beschreibbar, die Verbindung der verschiedenen Ebenen ist nicht möglich. Darüber hinaus sind an der Erbringung eines IT-Dienstes wie beschrieben verschiedene Personen beteiligt, die für ihre jeweilige Ebene verantwortlich sind, aber kein Detailwissen über die anderen Ebenen besitzen[26].

Unser Modell soll die Modellierung der Beziehungen und Abhängigkeiten zwischen den verschiedenen Ebenen eines IT-Dienstes ermöglichen. Es soll dazu die Informationen und eventuell vorhandene Spezialmodelle auf den jeweiligen Ebenen integrieren können. Dabei muss es zum einen die technischen Aspekte des IT-Dienstes abbilden können, muss dies aber einer für den Geschäftsprozessverantwortlichen verständliche Weise tun, ohne ihn mit technischen Implementierungsdetails zu überlasten. Es trägt damit zu einer Vereinheitlichung von Geschäftssicht und Technologiesicht, wie in [IBM06] gefordert, bei. Darüber hinaus soll es mit unserem Modell möglich sein, die Auswirkung von Ausfällen von Komponenten zu beschreiben. Aufbauend auf dieser Idee und den momentan eingesetzten Techniken, die wir im vorigen Kapitel beschrieben haben, werden wir in diesem Kapitel die Anforderungen an unser Modell ableiten.

Wir stellen dazu nochmals ausführlich die Anforderungen an ein gemeinsames Modell eines IT-Dienstes der verschiedenen Beteiligten an der Erbringung von IT-Diensten vor. Darüber hinaus stellen wir die Rahmenbedingungen für ein entsprechendes Modell vor.

3.1 Ebenen eines IT-Dienstes und deren Anforderungen

Wir haben im vorigen Kapitel ITIL und die verschiedenen existierenden Techniken für das Management von IT-Diensten beschrieben. Dabei sind wir auch bereits auf die unterschiedlichen Ebenen Geschäftsprozess, Software und IT-Infrastruktur eines IT-Dienstes und die verschiedenen Anforderungen, die sich aus diesen Ebene ergeben, eingegangen.

In diesem Abschnitt beschreiben wir die verschiedenen Personen oder Rollen, die an der Erbringung eines IT-Dienstes beteiligt und für die verschiedenen Ebenen verantwortlich sind und die wir teilweise bereits bei den ITIL Prozessen eingeführt haben. Darüber hinaus leiten wir deren Anforderungen an die Informationen über einen IT-Dienst ab, die sie für dessen technisches Management verwenden, oder die sie zur Beschreibung oder zur Dokumentation des Geschäftsprozesses, der durch den IT-Dienst realisiert wird, benötigen.

Wir definieren dabei für jede Ebene einen Verantwortlichen, unterscheiden zwischen:

- Geschäftsprozessverantwortlicher, als Verantwortlicher auf fachlicher Ebene
- Softwareentwickler, als Verantwortlicher für die Anwendungssysteme und die Softwarekomponenten
- Administrator, als Verantwortlicher für die IT-Infrastruktur, neben den IT-Systemen auch die physikalische Vernetzung der Systeme

Eine mögliche andere Aufteilung wäre, hier noch verschiedene Rollen auf den Ebenen zu unterscheiden. Auf der Ebene der Software wären dies zum Beispiel die Rollen Softwarearchitekt, Entwickler, Projektleiter und Tester. Es sind darüber hinaus noch weitere Rollen denkbar, ähnliches gilt auch für die anderen Ebenen. Da die Abgrenzung der Rollen voneinander schwierig ist

[26] Der Spezialfall, dass diese Rollen auf eine Person zusammenfallen ist nur für sehr kleine Szenarien vorstellbar, er widerspricht auch letztlich dem Konzept der Arbeitsteilung im Unternehmen

und sich Anforderungen an ein Modell teilweise aus mehreren Rollen ergeben und für die Aufstellung der Anforderungen eine weitere Aufteilung der Rollen nicht notwendig ist, verzichten wir auf die Aufteilung der Rollen in den einzelnen Ebenen. Die Aufteilung der Ebenen spiegelt vor allem die fachlich sehr leicht abgrenzbaren Gebiete wieder. Innerhalb der Ebenen ist teilweise auch unterschiedliches Detailwissen notwendig, ein gemeinsames Verständnis ist aber auf den jeweiligen Ebenen vorhanden.

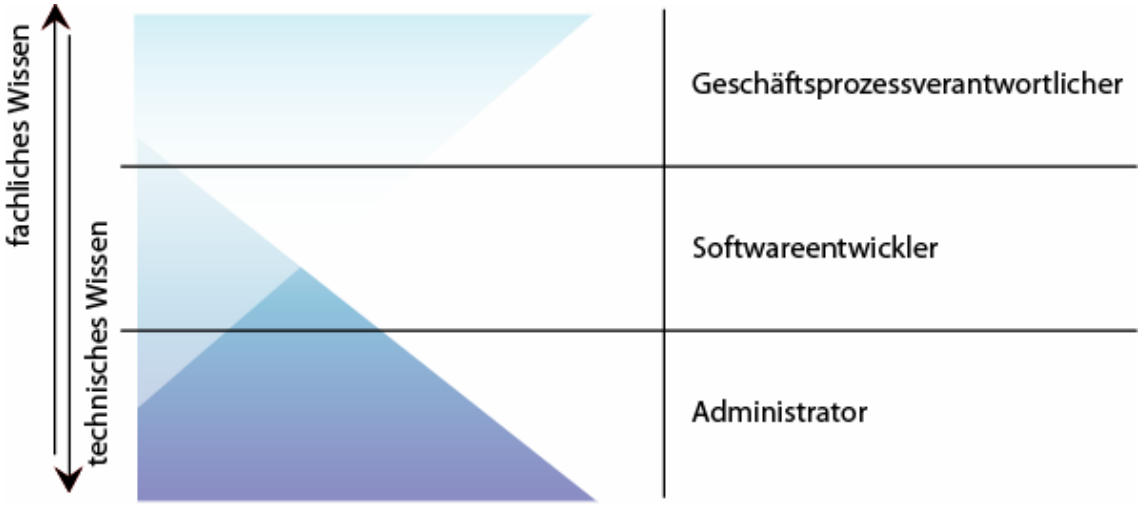

Abbildung 24 – Verantwortliche der Ebenen eines IT-Dienstes

3.1.1 Geschäftsprozessverantwortlicher

Der Geschäftsprozessverantwortliche verwaltet und beschreibt einen IT-Dienst auf der fachlichen Ebene. Er repräsentiert die Fachabteilung, die diesen Dienst im Unternehmen anbietet oder selbst verwendet. Er beschreibt die fachlichen Vorgaben für einen IT-Dienst und beschreibt den Dienst an sich. Dazu werden sowohl die internen Abläufe des Geschäftsprozesses beschrieben, als auch die Beziehungen zu anderen Geschäftsprozessen.

Der Geschäftsprozessverantwortliche trägt diese fachlichen Informationen zu einem Gesamtmodell eines IT-Dienstes bei. Für dieses werden auch Informationen benötigt, wie der Dienst erbracht wird. Dazu muss eine Abbildung des Geschäftsprozesses auf die Softwarekomponenten und die IT-Infrastruktur im Modell hinterlegt sein. Diese Abbildung wird zum einen mit dem Softwareentwickler bei der Einführung neuer Software oder neuer Funktionen erstellt und ergibt sich zum anderen, wenn diese Softwarekomponenten auf der IT-Infrastruktur eingerichtet werden. Die Kommunikation zwischen dem Softwareentwickler und dem Geschäftsprozessverantwortlichen erfordert dazu ein gemeinsames Modell, mit dem die beiden Ebenen und ihre Beziehungen auf der Basis von Funktionen möglichst einfach beschrieben werden können.

Es ergeben sich also folgende Anforderungen an ein Modell eines IT-Dienstes:

- Modellierung des Geschäftsprozesses zur Beschreibung des Geschäftsprozesses und seiner (internen) Abläufe.
- Modellierung der Beziehungen des betrachteten Geschäftprozesses zu Abläufen anderer Geschäftsprozesse.
- Möglichkeit der Beschreibung der benutzten und bereitgestellten Funktionen des Geschäftsprozesses zur Vereinfachung der Abbildung auf Softwarekomponenten und Entwicklung eines gemeinsamen Modells mit dem Softwareentwickler.
- Abbildung auf die Softwarekomponenten und auf die Hardware, auf der der Dienst ausgeführt wird, zur Überprüfung der Konformität der Ebenen Software- und Hardware mit rechtlichen Vorgaben und zum Abgleich, welche zusätzlichen Abhängigkeiten zu anderen Geschäftsprozessen sich durch die IT-Infrastruktur ergeben.

Wichtig ist, dass der Geschäftsprozessverantwortliche kein Experte auf der Ebene der Software und der Hardware ist. Das Modell muss zum einen die technischen Einschränkungen so beschreiben, dass Abweichungen oder Fehler sichtbar werden, darf den Geschäftsprozessverantwortlichen aber nicht mit Implementierungsdetails verwirren. Wo möglich müssen diese Informationen abstrahiert werden.

Wir haben bereits in der Einleitung beschrieben, dass wir kein vollständiges Weltmodell erstellen möchten, da dies zu einem nicht mehr handhabbaren Modell führen würde. Deswegen ergibt sich eine weitere Anforderung, die wir auch auf den anderen Ebenen wieder finden werden. Der Geschäftsprozessverantwortliche braucht für die Beschreibung seiner Geschäftsprozesse eine Möglichkeit, den Geschäftsprozess an sich mit seinen Abläufen zu modellieren. Dazu gehören über die Anforderungen für einen IT-basierten Dienst hinaus auch die Möglichkeit, Abläufe beschreiben zu können, die nicht als IT-Dienst in Software implementiert werden, sondern von einem Menschen durchgeführt werden. Der Geschäftsprozessverantwortliche benötigt also ein Modell zur umfassenden Geschäftsprozessmodellierung. Eine Möglichkeit dazu ist, wie schon eingeführt, ARIS, eine andere das Modell von Thurner [Thu04]. Wird eine entsprechende Modellierungstechnik verwendet, ist eine zusätzliche Anforderung an das Gesamtmodell:

- Möglichkeit zur Integration von speziellen Modellen der Ebene der Geschäftsprozesse, zum Beispiel ARIS, in das IT-Dienstmodell.

Diese Anforderung ist vor allem dann wichtig, wenn bereits ein entsprechendes Modell vorhanden ist. Diese Anforderung sorgt aber auch dafür, dass unser Teilmodell kein umfassendes Werkzeug zur Geschäftsprozessmodellierung ist und entsprechend einfach handhabbar bleibt. Es bietet damit einen einfachen Einstieg in die Geschäftsprozessmodellierung, falls hier noch kein umfassendes Modell existiert. Im Sinne einer umfassenden Geschäftsprozessmodellierung ist aber der Einsatz anderer Modellierungstechniken und die Integration dieser vorzuziehen.

3.1.2 Softwareentwickler

Auf der Ebene der Software ist der Softwareentwickler verantwortlich für den Entwurf von Softwarekomponenten oder die Adaption von Kaufsoftware zur Realisierung eines Geschäftsprozesses auf der IT-Infrastruktur. Er bekommt dazu Vorgaben vom Geschäftprozessverantwortlichen, die er in Software umsetzt und diese dem Administrator zur Installation und zum Betrieb auf der IT-Infrastruktur übergibt.

Der Softwarenentwickler trifft dabei verschiedene Entscheidungen, unter anderem wie die zu erstellende Software aufgebaut ist und welche Basiskomponenten sie verwendet. Diese Entscheidungen hängen von Informationen über die zukünftige Laufzeitumgebung der Anwendung ab. Dazu gehören Informationen über die vorhandene IT-Infrastruktur und die bereits installierten Softwarekomponenten und Dienste, die sich eventuell zur Vereinfachung der Entwicklung und zur Bildung einer homogenen Landschaft einbinden lassen. Neben diesen Entscheidungen über die Verwendung vorhandener Komponenten ist auch das Wissen über die momentane Auslastung der Infrastruktur notwendig, um die Laufzeit und Geschwindigkeit der eigenen Anwendung abschätzen zu können.

Es ergeben sich folgende Anforderungen an ein Modell eines IT-Dienstes:

- Beschreibung der Funktionen des Geschäftsprozesses, die implementiert werden müssen. Dazu kann unser Geschäftsprozessmodell dienen, um die Geschäftsprozesse auf die Softwarekomponenten abzubilden und eine gemeinsames Verständnis mit dem Geschäftsprozessverantwortlichen entwickeln zu können.
- Modellierung der zu erstellenden oder zu adaptierenden Software zur Beschreibung der Anwendung, ihrer internen Struktur und der Beziehung zu anderen Softwarekomponenten. Wie bei der Geschäftsprozessebene ergibt sich hier die Anforderung, existierende Spezialmodelle dieser Ebene, wie UML Diagramme, integrieren zu können.
- Modellierung der bereits existierenden Softwarekomponenten in der IT-Infrastruktur um Basiskomponenten und Dienste wieder verwenden zu können und so eine homogene

Softwarelandschaft aufzubauen sowie die Entwicklung neuer Softwareinseln und doppelt vorhandener Basisdienste zu vermeiden.

- Wissen über die Auslastung von Basissystemen, um eventuell steigende Anforderungen durch die eigene Software und deren Auswirkungen abschätzen zu können. Abschätzen der Laufzeit der entwickelten Software in der Laufzeitumgebung. Modellierung der Beziehungen zu den Basiskomponenten, um Auswirkungen bei der Einführung der neuen Softwarekomponenten beschreiben zu können.
- Wissen über die eingesetzten Managementframeworks, um eine Anbindung der eigenen Software zur Verwaltung und zur Überwachung durch das Managementsystem realisieren zu können. Modellierung dieser Frameworks, um ein Integrationskonzept entwickeln zu können.
- Wissen über die Laufzeitumgebung der entwickelten Software zur Fehleranalyse und Optimierung, wenn Fehler im Betrieb auftreten, im Sinne eines Software Reegineerings oder eines iterativen Softwareentwicklungsprozesses, der auch den Betrieb beinhaltet.
- Modellierung der beteiligten IT-Systeme und der Konfiguration der Basiskomponenten, um Fehler zu analysieren oder Auswirkungen von Änderungen an diesen Komponenten für die eigene Software bestimmen zu können, insbesondere bei Sicherheitsupdates oder neuen Versionen von Basiskomponenten.
- Entwicklung einer möglichst realistischen Testumgebung auf der Basis des Modells der produktiven Laufzeitumgebung.

Die Anforderungen des Softwareentwicklers beziehen sich neben der Notwendigkeit der Modellierung der Softwareebene auch auf die Ebene der Geschäftsprozesse und der IT-Infrastruktur.

Diese Ebene stellt die Klammer zwischen den beiden Ebene dar und ihr kommt in gewisser weise eine Vermittlerrolle zwischen den beiden Polen eines IT-Dienstes zu.

3.1.3 IT-Infrastruktur - Administrator

Auf der Ebene der IT-Infrastruktur ist der Administrator für den Betrieb der IT-Infrastruktur verantwortlich. Wie bereits in der Einleitung beschrieben, könnte man auch auf dieser Ebene unterschiedliche Rollen unterscheiden. Wie bereits ausgeführt verzichten wir darauf, da sich die Rollen überschneiden und die Anforderungen nicht eindeutig an einzelne Rollen zugewiesen werden können.

Wir haben im Abschnitt über ITIL bereits viele der Aufgaben des IT-Managements und damit des Administrators beschrieben. Zusammengefasst betreibt und wartet der Administrator die IT-Infrastruktur. Zu dieser gehören nicht nur die IT-Systeme, also Server mit ihren Komponenten wie Speichersubsystemen, sondern auch die Netzwerkinfrastruktur wie Ethernetnetze und Speichernetze (SAN's) und die daran angeschlossenen Speichersysteme.

Für den Administrator ist ein Modell seiner Infrastruktur wichtig, um diese zu planen, Fehler zu erkennen und den reibungslosen Betrieb aller Komponenten und der dadurch realisierten Softwaresysteme sicherzustellen. Dazu gehört sowohl die Überwachung der verfügbaren Dienste, als auch die Durchführung und Überwachung von Wartungsaufgaben wie der Sicherung der Daten. Ein Aspekt, wie bereits beschrieben, ist die Standardisierung der Infrastruktur. Dazu müssen verschiedene Implementierungen derselben Funktion gefunden und analysiert werden, um sie anschließend zu vereinheitlichen.

Es ergeben sich folgende Anforderungen an ein Modell eines IT-Dienstes:

- Modellierung aller IT-Systeme der IT-Infrastruktur mit den Komponenten, auf denen sie aufgebaut sind. Dazu gehört auch die Beschreibung der physikalischen Kommunikationsinfrastruktur.
- Modellierung der möglichen Kommunikationswege, durch die Modellierung von virtuellen Netzwerksegmenten (VLANs) auf der Ebene der Netze und der Auswirkung von Firewalls auf die Verbindungen. Gerade durch Techniken zur logischen Partitionierung

auf der Ethernet Ebene durch VLAN's oder aber auch der TCP/IP Ebene durch Subnetze erfordert eine feingranulare Modellierung, da die Verbindungen zwischen zwei IT-Systemen nicht auf die Existenz einer physikalischen Verbindung zurückgeführt werden können, wie im vorigen Kapitel bereits geschildert.

- Modellierung aller Applikationen und Softwarekomponenten mit ihrer Verteilung auf der IT-Infrastruktur. Modellierung der Beziehungen der Anwendungen und Softwarekomponenten untereinander, um die Auswirkungen beim Ausfall oder Neustart eines Servers auf die Gesamtinfrastruktur bestimmen zu können.
- Modellierung der Beziehungen zwischen Softwarekomponenten und Geschäftsprozessen, um diese Auswirkungen auch auf der Eben der Geschäftsprozesse bestimmen zu können.
- Simulation von Veränderungen an der Konfiguration von einzelnen Komponenten zur Bestimmung der Auswirkungen von Veränderungen. Root-Cause Analysis, um beim Ausfall einzelner Komponenten oder Dienste den Auslöser in der Infrastruktur identifizieren zu können.
- Analyse der IT-Infrastruktur im Bezug auf die Sicherheit, Auslastung und die Ausfallsicherheit von einzelnen Diensten. Planung von Vereinfachung der Infrastruktur durch die Eliminierung von Abhängigkeiten von verschiedenen Systemen voneinander durch die Zusammenlegung der Softwarekomponenten auf ein System.
- Informationen über den aktuellen Zustand alle Infrastrukturkomponenten in einem Leitstand zum Management der IT-Infrastruktur.
- Erkennung von unterschiedlichen Implementierungen derselben Funktionalität, um einen vorhandenen Bedarf zur Standardisierung zu erkennen. Beschreibung von Standards zur Vorgabe von Pattern bei der Planung neuer IT-Systeme.

Die Anforderungen des Administrators sind teilweise sehr stark technisch geprägt. Viele Informationen, wie die Monitoringdaten, die den Zustand aller Komponenten der IT-Infrastruktur anzeigen, sind heute durch die im vorigen Kapitel beschriebenen Werkzeuge bereits verfügbar. Es fehlt eine Abbildung auf die übergeordneten Ebenen des IT-Dienstes. Die Modelle auf der Ebene der IT-Infrastruktur sind ein guter Ausgangspunkt, da sie wesentlich detaillierte Informationen enthalten und teilweise, wie im vorigen Kapitel beschrieben, automatisch generiert werden können. Wichtig ist es, diese Modelle in eine vollständige Sicht integrieren zu können.

Veränderungen an der IT-Infrastruktur sind deutlich häufiger als Veränderungen in einem Geschäftsprozess[27]. Auch deswegen haben sich hier bereits die entsprechenden Monitoringwerkzeuge über die letzten Jahre entwickelt und verfeinert. Diese lassen sich nun, mit den Informationen über die Softwarekomponenten und die dadurch realisierten Geschäftsprozesse zu einem Gesamtbild oder einem Service-Dashboard erweitern.

3.2 Anforderungen an unser Modell

Aus den Herausforderungen für die IT-Infrastruktur und den Anforderungen der Beteiligten ergeben sich die Anforderungen an unser Modell und die Art der Modellierung, die wir im vorigen Abschnitt für die unterschiedlichen Ebenen bereits beschrieben haben.

In diesem Abschnitt fassen wir die Anforderungen der unterschiedlichen Ebenen nochmals zusammen und stellen darüber hinaus Anforderungen vor, die sich indirekt aus diesen Anforderungen ergeben.

[27] Änderungen bezeichnen hier grundlegende Veränderungen eines Dienstes in seiner Struktur und keine Variationen eines Geschäftsprozesses, wie die Anpassung eines Umrechungskurses, da diese Veränderungen bereits zur den geplanten Adaptionen des Geschäftsprozesses gehören (sollten).

Wir unterscheiden dabei fachliche Anforderungen, die die fachlichen Informationen beschreiben, die innerhalb des Modells abgebildet werden müssen und Anforderungen, die die Handhabung des Modells beschreiben.

Wichtig bleibt dabei, dass wir kein gemeinsames Modell im Sinne eines Weltmodells entwickeln möchten, sondern die einzelnen Ebenen aufeinander abbilden und zueinander in Beziehung setzen. Die einzelnen Ebenen sollen dazu abstrahiert werden und ein Modell entstehen, das auf allen Ebenen verständlich bleibt und die Implementierungsdetails verbirgt.

3.2.1 Fachliche Anforderungen

Die Fachlichen Anforderungen fassen die Anforderungen an die verschiedenen Modellelemente und die Mächtigkeit der Modellierung zusammen und beschreiben die Informationen, die das Modell enthalten muss, um den Anforderungen der unterschiedlichen Ebenen und ihrer jeweiligen Verantwortlichen gerecht zu werden.

Die fachlichen Anforderungen sind:

1. Modellierung von Geschäftsprozessen und ihrer gegenseitigen Abhängigkeiten, um Zusammenhänge zwischen den Geschäftsprozessen zu beschreiben.

   Es ist denkbar, dass zwei Geschäftsprozesse, die durch zwei völlig getrennte Softwaresysteme auf getrennten Servern realisiert werden, zueinander in Beziehung stehen[28]. Fällt eines der Softwaresysteme aus, wird die Abhängigkeit und die Beeinträchtigung des zweiten Geschäftsprozesses auf der technischen Ebene nicht deutlich. Dazu müssen die Abhängigkeiten, die sich nur auf der Ebene der Geschäftsprozesse ergeben, betrachtet werden können.

2. Modellierung der Struktur und der möglichen Abläufe von Softwaresystemen und damit Beschreibung der Beziehungen von Softwarekomponenten. Wichtig ist dabei auch die Modellierung der Beziehungen zu Basiskomponenten und bereits in der Infrastruktur vorhandenen Services. Darüber hinaus müssen Containerbeziehungen modelliert werden, um Softwarekomponenten, die in einer Laufzeitumgebung ausgeführt werden, modellieren zu können und diese Abhängigkeit so zu beschreiben.

3. Beschreibung der IT-Infrastruktur mit der physikalischen und der logischen Netzwerkinfrastruktur, um Fehler, die verschiedene Komponenten betreffen, analysieren zu können. Dazu gehört auch die Beschreibung von möglichen Abläufen und Firewallsystemen, im Gegensatz zur grobgranularen Modellierung wie in ARIS.

   Wir haben in den vorigen Abschnitten bereits beschrieben, dass die Existenz einer physikalischen Verbindung heute durch die logische Partitionierung von Netzen, durch Firewalls oder andere Sicherheitsmechanismen nicht mehr ausreicht, um eine logische Verbindung zwischen zwei Applikationen auf verschiedenen IT-Systemen zu beschreiben. Die Modellierung muss dem Rechnung tragen und die verschiedenen Netzwerkebenen in die Modellierung mit einbeziehen, vor allem auch die Anforderungen für die Modellierung virtueller Infrastrukturen. Entsprechende Beziehungen und Hierarchien, die sich für diese Infrastrukturen ergeben, müssen modellierbar sein. Darüber hinaus müssen diese Strukturen in einer Form dargestellt werden, mit der die technischen Implementierungsdetails versteckt werden.

4. Verbindung der Ebenen eines IT-Dienstes durch ein Modell, das die Ebenen Geschäftsprozess, Softwarekomponenten und IT-Infrastruktur verbindet und damit eine Modellierung aller Ebenen, die zu einem IT-Dienst beitragen.

[28] Dies ist zum Beispiel dann der Fall, wenn zwei Geschäftsprozesse durch den Benutzer miteinander verbunden sind, der die Daten von einer Anwendung in eine andere einträgt. Genau diese Beziehung wird auf der technischen Ebene dann nicht deutlich, da die Softwarekomponenten nicht miteinander verbunden sind. Das Bindeglied stellt in diesem Fall der Anwender dar.

5. Integration von existierenden, auf den jeweiligen Ebenen vorhandenen Modellen zur Modellierung der jeweiligen Ebene in die Modellierung der Beziehungen und Abhängigkeiten. Dadurch wird die Wiederverwendung bereits vorhandener Modelle erleichtert und die Verfeinerung von Teilmodellen auf den jeweiligen Ebenen, wie bereits beschrieben, ermöglicht.
6. Modellierung und Beschreibung der Auswirkung von Ausfällen, um nicht nur die Abhängigkeiten der Komponenten voneinander, sondern auch die Funktion von Komponenten zu modellieren. Damit ist die Beschreibung der Funktion eines Clusters möglich, der im Falle des Ausfalls eines physikalischen Knotens die Dienste auf einen anderen Knoten verlagert und somit der Ausfall keine direkten Konsequenzen auf die Dienstverfügbarkeit hat.
7. Erkennung von unterschiedlichen Implementierungen derselben Funktionalität, um einen vorhandenen Bedarf zur Standardisierung zu erkennen. Beschreibung von Standards zur Vorgabe von Pattern bei der Planung neuer IT-Systeme.

Diese fachlichen Anforderungen beschreiben die Anforderungen der unterschiedlichen Ebenen zur Modellierung eines IT-Dienstes mit all seinen Ebenen, den Beziehungen und Abhängigkeiten und den Konsequenzen eines Ausfalls einer Komponente auf die gesamten IT-Dienste eines Unternehmens.

Darüber hinaus ergeben sich aber auch Anforderungen, die den Umgang mit dem Modell betreffen.

3.2.2 Anforderungen an die Nutzbarkeit

Während die fachlichen Anforderungen die Informationen des Modells beschreiben, beschreiben wir in diesem Abschnitt die Anforderungen an die Handhabung des Models. Diese umfassen folgende Aspekte:

1. Intuitiv verständliches Modell

   Das Modell soll auch für einen Benutzer verwendbar sein, der wenige Kenntnisse in der Modellierung oder Architektur von Softwarekomponenten oder IT-Infrastrukturen hat. Es soll sich möglichst an den etablierten Standards der Modellierung auf den verschiedenen Ebenen orientieren und diese einfach integrieren können, um wie bereits beschrieben, die Wiederverwendbarkeit vorhandener Modelle sicherzustellen.

2. Einfache grafische Darstellung

   Unser Modell soll nicht die Softwarekartographie ersetzen oder eine Alternative dazu darstellen, die sich intensiv mit verschiedenen Sichten und graphischen Darstellungsformen für IT-Dienste beschäftigt.

   Trotzdem ist eine möglichst einfache Umsetzung des Modells in eine grafische Darstellung notwendig, um die Visualisierung und die Handhabung der Modelle zu vereinfachen und dem Nutzer eine einfache Form der Modellierung an die Hand zu geben. Gerade die Kommunikation der einzelnen Verantwortlichen miteinander wird dadurch vereinfacht. Das Modell soll möglichst einfach auch an einer Tafel oder auf Papier genutzt werden können.

3. Integration in ein Tool zum IT-Management

   Wir haben bereits in der Einleitung beschrieben, dass unsere Vision des IT-Managements ein Management von IT-Diensten auf einer fachlichen Ebene ist. Dazu benötigen wir entsprechende Tools, die die fachlichen Anforderungen, in entsprechende Richtlinien übersetzen und auf der IT-Infrastruktur umsetzen können.

   Darüber hinaus haben wir mit ITIL bereits existierende Tools rund um das IT-Dienstmanagement beschrieben. Das Modell soll möglichst einfach in entsprechende Tools integrierbar sein, eine Umsetzung in ausführbaren Code sollte möglichst einfach

erfolgen können. Dadurch ist es zum Beispiel möglich, in existierende Monitoringwerkzeuge sehr einfach die Geschäftsprozessebene zu integrieren.

4. Konsistenzsicherung durch das Modell

   Das Modell soll für sich selbst eine Konsistenzsicherung durchführen und sicherstellen, dass die Informationen innerhalb des Modells richtig und in einem konsistenten Zustand sind. Diese Konsistenzsicherung lässt sich ebenfalls sehr einfach in ein Tool, das unser Modell umsetzt, integrieren.

Diese Anforderungen sollen vor allem die leichte Anwendbarkeit unseres Modells sicherstellen.

3.3 Fazit

Wir haben in diesem Kapitel die Anforderungen an ein Modell zur IT-Dienstmodellierung zusammengefasst. Dabei sind wir auch auf die verschiedenen Ebenen und der beteiligten Personen eingegangen und haben die Anforderungen abgeleitet.

Im nächsten Kapitel werden wir unser Modell vorstellen und die Entscheidungen, die bei der Erstellung des Modells getroffen wurden, mit diesen Anforderungen begründen.

4 Konzepte und Möglichkeiten des Modells

Wir haben im vorigen Kapitel die Anforderungen an unser Modell entwickelt und beschrieben. In diesem Kapitel stellen wir die Grundkonzepte des Modells und die verschiedenen Teilmodelle vor und beschreiben die Abwägungen, die dabei getroffen wurden. Nach dieser Einleitung beschreiben wir im nächsten Kapitel die Teilmodelle detailliert und definieren die einzelnen Elemente formal.

4.1 Aufgabe des Modells

Diese Arbeit stellt ein abstraktes Modell zur IT-Dienstmodellierung vor, das die Modellierung von Abhängigkeiten auf den verschiedenen Ebenen eines IT-Dienstes sowie die Beschreibung der Modellebenen Geschäftsprozess, Softwarekomponenten und IT-Infrastruktur ermöglicht. Darüber hinaus dient ein Funktionsmodell zur besseren Strukturierung und macht verschiedene Implementierungen vergleichbar. Die wichtigsten Merkmale des Modells sind:

- Vereinfachte Teilmodelle auf den Ebenen Geschäftsprozess, Software und IT-Infrastruktur zur Beschreibung der Abhängigkeiten auf den jeweiligen Ebenen. Ein Funktionsmodell dient zur Strukturierung der Modelle und ermöglicht den Vergleich zwischen verschiedenen Implementierungen.
- Möglichkeit der Integration existierender Modelle auf den jeweiligen Ebenen.
- Modellierung der Beziehungen zwischen dem Funktionsmodell, dem Geschäftsprozessmodell, dem Softwaremodell und dem Infrastrukturmodell.
- Abstraktion der Details der Implementierung der verschiedenen Ebenen und vollständigen Modelle dieser Ebene zur einfacheren Handhabbarkeit durch Verantwortliche anderer Ebenen.
- Möglichkeit der Beschreibung der Konsequenzen von Ausfällen auf die Komponenten eines IT-Dienstes.

Das Modell ermöglicht so die Modellierung eines IT-Dienstes, die die fachliche Sicht und die IT-Infrastruktur miteinander verbindet, wie dies auch von den befragten CEOs der IBM Studie gefordert wird [IBM06]. Es kann als Basis für ein Managementsystem verwendet werden, das eine Gesamtsicht über die verschiedenen Ebenen und Komponenten eines IT-Dienstes bietet und dabei die jeweiligen Anforderungen und Abhängigkeiten darstellen kann. Darüber hinaus verbessert es das Impact Management und kann die Planung von IT-Landschaften deutlich vereinfachen. Wir gehen in einem späteren Abschnitt auf diese Möglichkeiten genauer ein.

Das Modell basiert auf verschiedenen Annahmen und Modellierungsentscheidungen, die wir im Folgenden vorstellen. Diese ergeben sich aus dem im vorigen Kapitel vorgestellten Anforderungen.

4.2 Die Modellierungstechnik

Eine wichtige Frage bei der Entwicklung unseres Modells ist die Frage, welche Modellierungstechnik und welches Basismodel verwendet werden soll, auf dem sich die Modellierung abstützt. Für die Modellierung stellt die Informatik sehr verschiedene Konzepte und Verfahren zur Verfügung, die sich in ihrer Komplexität und Mächtigkeit deutlich unterscheiden. Die Wahl der Modellierung ist dabei vor allem durch die Problemstellung beeinflusst. Modelle, die in der Praxis zum IT-Dienstmanagement eingesetzt werden, haben wir bereits in Kapitel 2.3.4 vorgestellt. Wir werden in diesem Abschnitt verschiedene Modellierungstechniken vorstellen und unsere Wahl des Basismodells begründen.

Unser Modell dient dazu, Zusammenhänge zwischen autonomen Einheiten zu beschreiben und die Verbindungen und Abhängigkeiten der Komponenten verschiedener Ebenen zu verdeutlichen. Darüber hinaus möchten wir eine Möglichkeit der grafischen Darstellung. Hieraus ergeben sich verschiedenen Möglichkeiten der Modellierung, wir werden einige kurz vorstellen.

Für unser übergeordnetes Ziel, der Modellierung und Beschreibung von Auswirkungen von Ausfällen von einzelnen Komponenten auf ein Gesamtsystem ergeben sich zwei Arten der Modellierung. Zum einen die Systemmodellierung, zum anderen die Modellierung von möglichen Fehlern des Systems. Durch beide Möglichkeiten der Modellierung lassen sich das Auftreten und die Wirkung von Ausfällen in einem System beschreiben.

Die Systemmodellierung beschreibt die Architektur eines Systems mit den verschiedenen Komponenten des Systems und deren Beziehungen zueinander. Es haben sich verschiedene Modellierungstechniken etabliert, von denen wir einige exemplarisch vorstellen und deren Möglichkeiten vergleichen:

- Graphentheorie

  Die Graphentheorie ist eine der elementarsten Modellierungstechniken der Mathematik und der Informatik, auf der viele andere Techniken basieren. Viele Probleme, die sich in der Informatik finden, lassen sich auf Graphen zurückführen und mit den Mitteln der Graphentheorie lösen. Darüber hinaus existieren für viele Graphentheoretische Probleme effiziente Algorithmen, um diese zu lösen.

  Graphen bestehen aus eine Menge an Knoten (oft auch Punkte oder Ecken bezeichnet), die durch Kanten miteinander verbunden sind. Diese lassen sich auch sehr einfach grafisch visualisieren. Graphen können verschiedene mathematische Eigenschaften besitzen, sie können endlich oder unendlich sein.

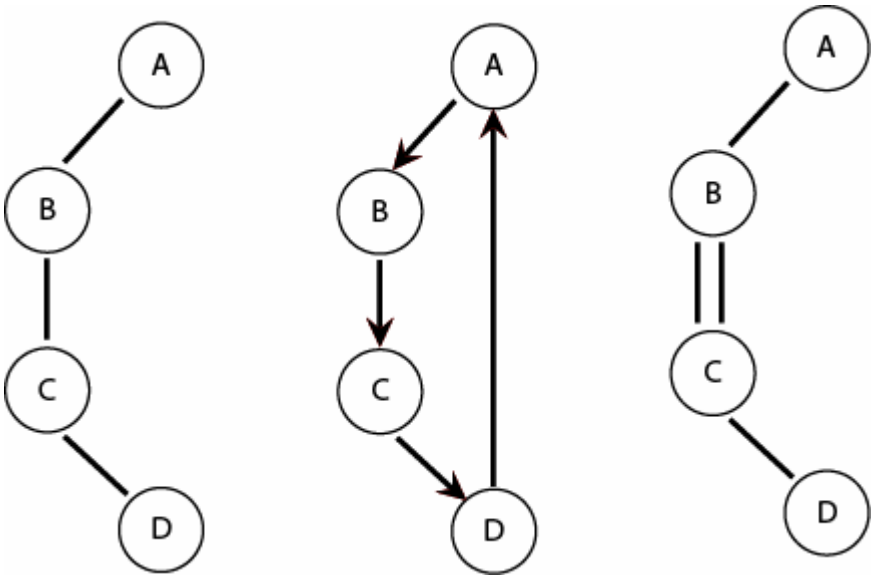

Abbildung 25 – Graphen: ungerichtet, gerichtet, ungerichtet mit Mehrfachkanten

Die Abbildung 25 zeigt drei unterschiedliche Typen von Graphen. Der erste Graph ist ein ungerichteter Graph mit vier Knoten und drei Kanten. Die Kanten sind in diesem Fall nicht gerichtet und beschreiben keinen Ablauf oder eine Rangfolge.

Der mittlere Graph erhält diese erweiterte Information. Seine vier Kanten sind gerichtet und stellen so einen Ablauf in diesem Graphen dar. Dieser Ablauf ergibt einen Kreis oder einen Zyklus. Diese Eigenschaft ist häufig ein wichtiges Merkmal für einen Graphen, genauer die Frage, ob ein Graph einen Zyklus enthält.

Der dritte Graph entspricht dem ersten Graphen, enthält jedoch Mehrfachkanten. In diesem Beispiel existieren zwischen dem Knoten B und dem Knoten C zwei Kanten. Die Anzahl der mehrfach vorhandenen Kanten ist beliebig.

Eine Spezialform von Graphen ist ein Wald. Ein Wald besteht aus ungerichteten Graphen ohne einen Kreis. Sind alle Graphen zusammenhängend, spricht man hier auch

von einem Baum. Durch entfernen der Komponente, die zwei Bäume verbindet, entsteht ein Wald.

- Petrinetze

  Petrinetze basieren auf Graphen[29], lassen jedoch Aussagen über mögliche und unmögliche Abläufe zu.

  „Petrinetze sind gekennzeichnet durch eine strenge Trennung von passiven und aktiven Systemkomponenten, dargestellt durch passive und aktive Netzelemente, die Stellen bzw. Transitionen genannt werden. Die passiven Stellen (Elemente der Menge S, auch oft mit P für places bezeichnet) repräsentieren (lokale) Zustände des Systems, die während des Ablaufs eines Systems erfüllt sein können.

  Transitionen (Elemente der Menge T) stellen Aktionen oder Ereignisse dar, für deren Ausführung bzw. Eintreten bestimmte Zustände Voraussetzung sind und nach deren Auftreten bestimmte Zustände erreicht sind. In Bezug auf Transitionen kann man die Zustände also auch als Bedingungen auffassen. Die Veränderung der Zustände des Systems durch eine Transition wird ihre Wirkung genannt und durch die sogenannte Flußrelation F codiert“

Definition 12 – Petrinetz (Quelle [GI-PN])

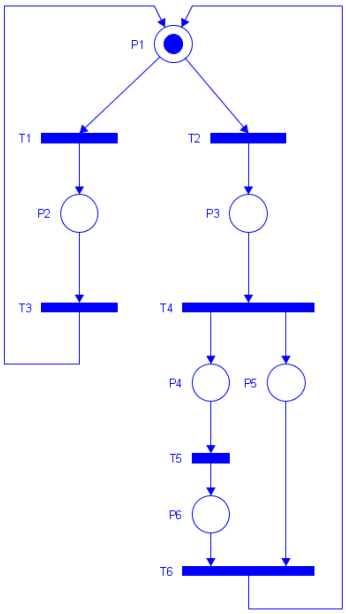

Abbildung 26 – Petrinetz (Quelle Wikipedia)

Die Abbildung 26 zeigt ein Beispiel für ein Petrinetz mit sechs Positionen. Der Startpunkt P1 ist mit einem Element vorbelegt.

- Entity Relationship Modellierung

  Die Entity Relationship Modellierung (ER) und die entsprechenden ER-Diagramme dienen dazu, Informationen und die Beziehungen zwischen diesen zu beschreiben. Sie lassen sich einfach dazu verwenden, Datenbanken zu definieren bzw. den Inhalt einer Datenbank zu beschreiben. Modelle, die auf dem ER-Modell basieren, lassen sich deshalb sehr leicht direkt in einer Datenbank speichern.

[29] Formal ist ein Petrinetz ein bipartiter, gerichteter Graph.

Die Informationen werden dabei in den Entitäten abgelegt. Beziehungen zwischen den Elementen werden in Relationen abgebildet. Die Abbildung 27 verdeutlicht dies an einem einfachen Beispiel: Zu den Mitarbeitern, die über einen Namen referenziert werden, werden hier Projekte mit einer Beschreibung gespeichert, die der jeweilige Mitarbeiter leitet. Die ER-Modellierung ist vor allem in der Softwareentwicklungsphase sehr weit verbreitet.

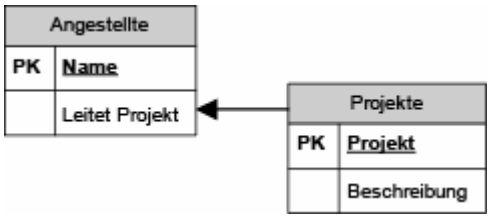

Abbildung 27 – Vereinfachtes ER-Diagramm

- Focus
  Eine weitere Technik ist die am Lehrstuhl von Prof. Manfred Broy entwickelte und auch in vielen kommerziellen Projekten erfolgreich eingesetzte Technik Focus [Broy, Stølen 01]. Focus wird momentan vor allem zur Modellierung eingebetteter Systeme eingesetzt.

  Focus basiert auf der Modellierung von Strömen zwischen verschiedenen Systemen und bietet sehr weitreichende Möglichkeiten zur Beschreibung des Systemverhaltens. Mit AutoFOCUS steht ein Tool zu Verfügung, das die Modellierung ermöglicht. Für AutoFOCUS existieren Integrationen in verschiedene Beweisersysteme, um Aussagen über Systeme überprüfen zu können und in Codegeneratoren, um direkt aus den Modellen ausführbaren Code erzeugen zu können.

Die beschriebenen Techniken unterscheiden sich vor allem in Ihrer Komplexität und ihrer Spezialisierung auf einzelne Problemstellungen. Gleiches gilt für die bereits in vorigen Kapiteln beschriebenen Techniken von ITIL und auch der UML, die ebenfalls die Systemstruktur modellieren.

Eine andere Art der Modellierung von Auswirkungen von Fehlern, die in einem System auftreten, ist die Beschreibung der Fehlerzustände selbst, die in diesem System auftreten können. Eingesetzt werden diese Arten der Modellierung unter anderem auch in der Risikobewertung von Kernkraftwerken. Dabei werden alle Fehlerzustände eines Systems und ihre Auswirkungen beschrieben und nicht das System als solches modelliert.

- Fehlerbaumanalyse

  „Bei der Fehlerbaumanalyse handelt es sich um eine Top-Down-Methode, bei der Fehler eines Systems beschrieben werden. Man gibt ein Ereignis vor, das nicht eintreten soll, das sogenannte Top-Ereignis, und beschreibt in Form einer Baumstruktur, welche untergeordneten Ereignisse wie eintreten müssen, damit das jeweilige übergeordnete Ereignis eintritt. Man stellt dabei ein so genanntes Fehlerbaummodell auf. Die Wahl der Systemsicht im Modell hängt dabei von der speziellen Fragestellung, dem Top-Ereignis, ab. Als Verknüpfungselemente stehen in der Basisform UND-Gatter und ODER-Gatter zur Verfügung..

  Definition 13 – Fehlerbaumanalyse (Quelle [ViSEK])

  Die Fehlerbaumanalyse eignet sich dabei vor allem für Probleme, bei denen alle weiteren Konsequenzen eines Ereignisses bekannt und beschreibbar sind.

- Ereignisbaumanalyse

  „Bei der Ereignisbaumanalyse (Event Tree Analysis, ETA) handelt es sich um eine induktive Methode zur Entwicklung der möglichen Folgen eines auslösenden Events, zum Beispiel eines Fehlers. Die Folgen eines solchen Events können einer-

seits entschärft werden, andererseits aber auch verschlimmert werden, indem sie unmittelbar nach ihrem Auftreten von Systemkomponenten verarbeitet werden. Der ETA liegen sog. Markov Ketten zugrunde.“

Definition 14 – Ereignisbaumanalyse (Quelle [ViSEK])

Bei der Fehlermodellierung müssen alle Ereignisse oder Fehler modelliert werden, um Aussagen über Ausfälle machen zu können. Die Beziehungen werden dabei implizit durch die Fehler modelliert. Dabei kommt es immer wieder vor, dass Abhängigkeiten vergessen werden, oder Fehler auftreten, an bei der Modellierung nicht betrachtet wurden.

Die Systemmodellierung ist im Gegensatz dazu wesentlich näher an der Vorstellung eines Administrators des Systems, das er „vor sich sieht“. Wir verwenden deswegen einen Ansatz zur Systemmodellierung.

Da wir ein möglichst einfaches Modell und eine grafische Darstellung benötigen, werden wir uns auf die Graphentheorie abstützen. Petrinetze stellen durch die Modellierung möglicher Abläufe bereits Funktionen zur Verfügung, die wir nicht benötigen, da wir keine konkreten Abläufe, sondern nur Beziehungen zwischen den Komponenten beschreiben möchten und keine Aussagen über die tatsächlichen Abläufe treffen.

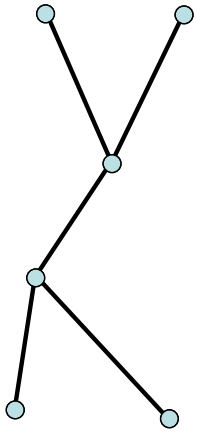

Abbildung 28 – Grundstruktur unseres Modells

Ansätze wie FOCUS sind nochmals deutlich mächtiger und komplexer, sie stellen uns einen Funktionsumfang zur Verfügung, den wir nicht ausschöpfen. Darüber hinaus hat sich gerade im Umfeld von FOCUS gezeigt, dass komplexe Modellierungstechniken eine große Hürde für den Einstieg in ein Modell sein können und deren Einsatz erschweren. Die Entity Relationship Modellierung käme den Modellen der CMDBs sehr nahe[30], beinhaltet für sich aber keine grafische Modellierungstechniken, für die Inhalte, die in den Relationen abgelegt werden. UML wird sehr stark in der Modellierung und Entwicklung von Software eingesetzt. Die UML hat aber deutliche Schwächen bei der Modellierung der IT-Infrastruktur und erschließt sich Verantwortlichen anderer Ebenen nicht immer direkt.

Die Abbildung 28 verdeutlicht die Grundstruktur unseres Modells. Die Knoten repräsentieren dabei Elemente der jeweiligen Ebene. Die Kanten beschreiben die Beziehungen und die Abläufe zwischen diesen. Wir werden bei der Vorstellung der einzelnen Modellebenen noch genauer auf die jeweils verwendeten Graphen eingehen.

[30] CMDBs basieren meist auf relationalen- oder objektorientierten Datenbanken.

4.3 Vorgehensweise zur Modellerstellung

Wie schon in den vorigen Kapiteln vorgestellt, gibt es verschiedene Ausgangspunkte für die Erstellung eines Modells eines IT-Dienstes, das die verschiedenen Ebenen integriert. Wir gehen in diesem Abschnitt auf unser Vorgehensmodell ein, vergleichen dieses mit den Top-Down Ansätzen von Thiele [Thiele05] und der Softwarekartographie und beschreiben die Besonderheit der Integration der Softwareentwicklung in die Erstellung eines Dienstmodells.

Gerade auf der Ebene der IT-Infrastruktur ist oft eine gute Toolunterstützung vorhanden, mit der Modelle über den Ist-Zustand der IT-Infrastruktur erstellt werden können. Diese Tools umfassen meist sowohl die Hardwarelandschaft, als auch die Softwareebene und die Verteilung der Softwarekomponenten auf den verschiedenen Ebenen. Wir haben hier die ITIL Tools CMDB und Application Scanner bereits eingeführt. In diese Managementsysteme sind teilweise auch Modelle der Anwendungssysteme integriert, die die Abhängigkeiten zwischen den Softwarekomponenten für einzelne Anwendungen kennen und diese so in ein Modell der IT-Dienste auf der Softwareebene integrieren und darstellen können. Ein Beispiel ist hier der in Kapitel 2.3.3 eingeführte Microsoft Operations Manager mit seinen Management Packs. Diese Modelle können als Basis für einen Bottom-Up orientierten Ansatz zur Erstellung der Modelle eines IT-Dienstes verwendet werden.

Auf der Ebene der Geschäftsprozesse werden häufig nur die Geschäftsprozesse an sich mit ihren Abläufen und Abhängigkeiten modelliert. Die Beziehung zur Software wird meist nicht beschrieben. Konzepte, wie sie zum Beispiel die Softwarekartographie zur Verfügung stellt, wollen dies ändern. Damit wird die Softwarelandschaft mit in die Modelle integriert und es entstehen Karten, die die Softwarelandschaft und teilweise die IT-Infrastruktur beschreiben. Diese Modelle verfolgen eher einen Top-Down Ansatz zur Erstellung eines IT-Dienstmodells. Ein Extrembeispiel ist der von Thiele vorgestellte und bei einigen Bereichen der BMW AG verfolgte Ansatz, auf den wir im folgenden Abschnitt eingehen.

4.3.1 Top-Down Modellierung nach Thiele

Wir haben in Kapitel 2.3.5 schon den Ansatz von Thiele [Thiele05] kurz vorgestellt, der so auch momentan bei BMW geplant wird. Der Ansatz sieht eine zentral gesteuerte Planung der Infrastruktur mit Managern vor, die ausgehend von den Geschäftsprozessen die Softwarelandschaft und die IT-Infrastruktur planen. Die IT-Infrastruktur wird hier nachgelagert eingebunden und kann durch Platzhalterobjekte sehr lange auch aus der Modellierung herausgelassen werden. Aber auch die IT-Infrastruktur wird dabei von entsprechenden Managern einer Planungsabteilung und nicht der IT-Abteilung entworfen.

Wir betrachten diesen Ansatz als nicht tauglich und eine reine Fokussierung auf einen Top-Down Ansatz für nicht zielführend. Gerade große Veränderungen in der IT-Landschaft, wie sie nach einer Fusion oder Integration einer Unternehmung in eine andere auftreten, sind von eine sehr großen Dynamik sowie hohem Zeitdruck geprägt. Die dabei entstehenden Integrationen zwischen den Geschäftsprozessen finden am Anfang meist auf der Ebene der IT-Infrastruktur und der Softwarekomponenten statt. Der Entwurf gemeinsamer Geschäftsprozesse und die Homogenisierung der IT-Infrastruktur ist meist ein nachgelagerter Prozess.

Ein Beispiel hierfür ist die Fusion von Compaq und HP. Beide Firmen betrieben die zur damaligen Zeit größten Microsoft Exchange Installationen, die verteilt auf insgesamt ca. 450 Server waren[31]. Darüber hinaus waren die Struktur der Installation sowie die eingesetzten Versionen unterschiedlich. Eine der Vorgaben war, am ersten Tag der gemeinsamen Arbeit ein einheitliches Emailverzeichnis bereitzustellen. Diese recht einfache und notwendige fachliche Vorgabe stellte bei unterschiedlichen Systemen und bereits für sich jeweils einzeln betrachteten komplexen Strukturen eine Herausforderung dar, da auch entsprechende Tools von Microsoft nicht zur

[31] Microsoft Exchange ist ein Mailsystem, das sich aus mehreren Softwarekomponenten zusammensetzt und ein sehr gutes Scale-out Verhalten aufweist. Entsprechende Infrastrukturen neigen deshalb dazu, sehr viele Server zu umfassen, auch um die Zeit zur Systemwiederherstellung beim Totalausfall möglichst gering zu halten.

Verfügung standen und von Microsoft unter anderem ausgehend von diesem konkreten Projekt entwickelt wurden. Eine Homogenisierung der Abteilungen und Emailstruktur wurde bei der neuen HP nachgelagert durchgeführt.

Interessant und wichtig ist an diesem Beispiel, dass zur Umsetzung einer sehr konkreten und auf den ersten Blick einfachen fachlichen Vorgabe in diesem Fall ein detailliertes Wissen über die Struktur der existierenden Systeme notwendig ist, um überhaupt eine Verbindung und Integration planen zu können [MSEX04]. Ausgehend von einer Verbindung zwischen den beiden Welten und regelmäßiger Synchronisation der beiden Systeme wurde erst über einen längeren Zeitraum eine einheitliche Struktur geschaffen. Selbst zwei Jahre nach der Übernahme sind heute immer noch Systeme im Einsatz, die einer der beiden früheren Installationen zuzuordnen sind.

Ein weiterer wichtiger Faktor, der gegen einen rein Top-Down getriebenen Ansatz spricht ist, dass auf den verschiedenen Ebenen vertikale Abhängigkeiten zwischen Komponenten existieren, die nicht Teil der Prozesse auf höheren Ebenen sind. Infrastrukturdienste wie Benutzerverwaltung, Sicherheit und Containerbeziehungen zwischen Softwarekomponenten werden erst auf der jeweiligen Ebene deutlich, auf der sie tatsächlich relevant sind und sind nicht Bestandteil der Geschäftsprozessmodellierung. Ein Beispiel dafür ist die Einbettung von Softwaresystemen in die schon beschrieben Active Directory (AD) Infrastruktur von Windows. Softwarekomponenten, die ihre Benutzerinformationen im AD ablegen, benötigen Server, die in die AD Struktur eingebunden sind und stellen dahingehend Anforderungen an die Platzierung des Servers, auf dem sie ausgeführt werden in der Infrastruktur, was wieder Konsequenzen für die Sicherheit der Gesamtinfrastruktur haben kann. Diese Einschränkungen werden erst bei einer umfassenden Betrachtung deutlich und gerade das Beispiel von AUDI mit dem Blackberry zeigt die Konsequenzen, wenn diese Betrachtung nicht erfolgt.

Die Platzhalterobjekte zur vereinfachten Modellierung, wie sie Thiele vorstellt, erscheinen deswegen als sehr kritisch. Werden Beziehungen zu anderen Systemen oder Basisdiensten bei der Planung durch die Verwendung von vereinfachenden Platzhalterobjekten zu lange nicht beachtet, können Soll-Planungen entstehen, die mit der konkreten IT-Infrastruktur nicht umsetzbar sind. Die Entwicklung eines vollständigen Modells eines IT-Dienstes erfordert Fachwissen über alle Ebenen der Erbringung. Die vorgeschlagenen Manager, die abstrakt Vorgaben für die Entwicklung machen und diese an die Verantwortlichen der einzelnen Ebenen weitergeben, scheinen nicht zielführend und führen zu Medienbrüchen, die einer umfassenden und homogenen Modellierung entgegenstehen. Gerade wenn dieser Ansatz zu Management einzelner „Silos“ verwendet wird, wir die Standardisierung auf einzelnen Ebenen eventuell nicht berücksichtig. Darüber hinaus können durch sich abzeichnende neue Versionen von Software heute Implementierungen notwendig sein, die auf diese Veränderungen vorbreiten und momentan komplexer als notwendig sind. Wird eine reine Top-Down orientierte Planung und Optimierung verwendet, ist dieses Wissen eventuell nicht vorhanden.

4.3.2 Integrierte Modellierung

Die Abbildung 2 aus Seite 6 verdeutlicht die verschiedenen Arten von Wissen, die über die verschiedenen Ebenen eines Dienstes notwendig sind. Unserer Ansicht wird diesem Umstand mit dem Ansatz von Thiele zu wenig Bedeutung beigemessen. Die Arbeit setzt entweder voraus, dass ein Modell von einem Experten auf allen drei Ebenen erstellt werden, was wir für schwierig halten, oder aber das erstellte Modell beachtet Implementierungsdetails nicht, was zu den schon im vorigen Kapitel beschriebenen Konsequenzen führen kann. Darüber hinaus ist es möglich, dass statt einer Standardisierung der IT-Dienste eher noch mehr Einzellösungen entstehen, wenn Vorgaben „mit Gewalt“ umgesetzt werden, um diese zu erfüllen.

Wir betrachten einen integrativen Ansatz, bei dem alle Beteiligten der drei Ebenen zusammen an einem gemeinsamen Modell eines IT-Dienstes arbeiten und diesen, auf ihren jeweiligen Ebenen, entwickeln, indem sie die Möglichkeiten und Modelle auf der jeweiligen Ebene einsetzen, für zielführender. Die Experten der jeweiligen Ebenen können so ein gemeinsames Verständnis entwickeln und ihre jeweiligen Sichten in das Gesamtmodell integrieren. Entsprechend stellt unser Modell die Verbindung zwischen diesen drei Ebenen her und ermöglicht so die Er-

stellung eines konsistenten IT-Dienstmodells, ohne die Beteiligten auf den jeweiligen Ebenen einzuschränken oder von Ihnen Detailwissen über die anderen Ebenen zu erwarten.

Wichtig ist in diesem Zusammenhang auch, dass in diesem Schritt ein Modell eines einzigen IT-Dienstes entwickelt wird. Dieses Modell besteht aber nicht für sich selbst und ein Unternehmen wird sicher auch nicht nur einen IT-Dienst ausführen. Es ist also wichtig, dass durch unser Modell und der Modellierung einzelner IT-Dienste ein Gesamtmodell aller IT-Dienste eines Unternehmens entsteht.

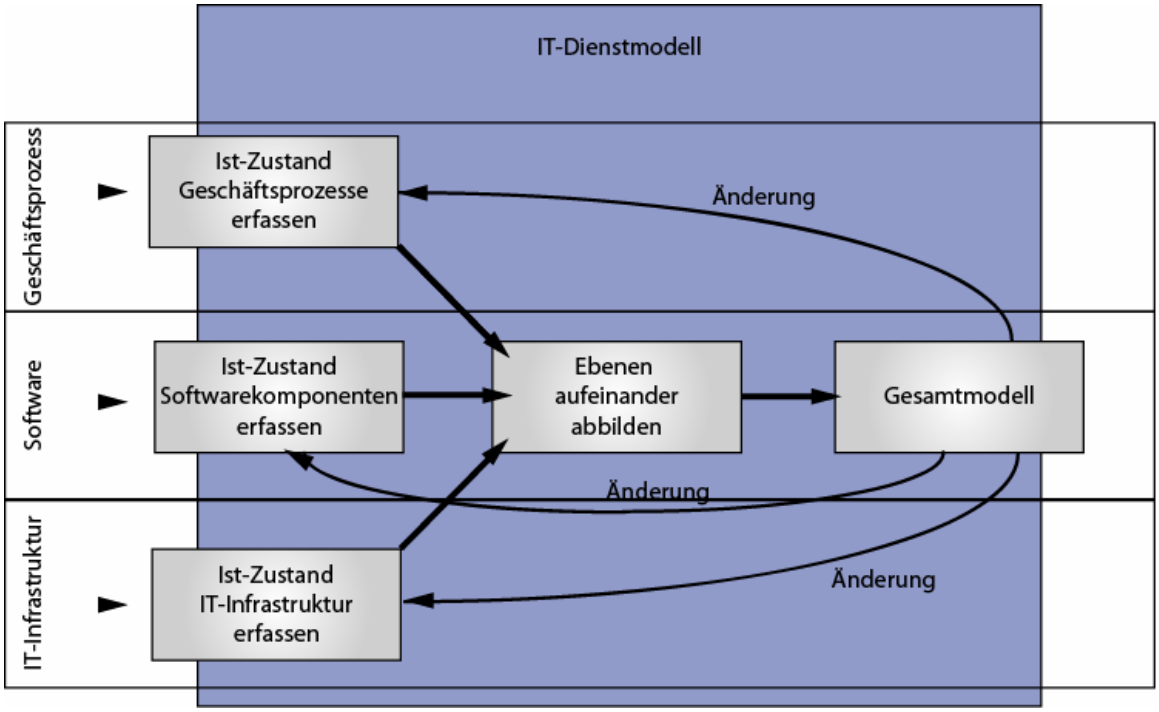

Abbildung 29 – Unser Modell im allgemeinen Prozess zur Erstellung eines IT-Dienstmodells

Wir haben bereits im Kapitel 2.3.2 ein allgemeines Modell zur Erstellung eines IT-Dienstes eingeführt. Die Abbildung 29 ordnet unser IT-Dienstmodell in diesen Prozess ein. Auf den einzelnen Ebenen wird der Ist-Zustand oder die geplanten Geschäftsprozesse, Software- und die Infrastrukturkomponenten erfasst und entweder mit den jeweiligen Spezialmodellen der Ebene oder mit einem unserer Teilmodelle beschrieben. Diese werden dann aufeinander abgebildet und es entsteht so ein Gesamtmodell, das aus den Teilmodellen und den Verbindungen zwischen diesen besteht.

Im Extremfall, wenn die bereits beschriebenen Managementtools eingesetzt werden, die Informationen auf der Softwareebene zu den erfassten Daten über die IT-Infrastruktur hinzugefügt werden und durch die Softwarekomponenten dann eine Abbildung auf die Geschäftsprozesse durchgeführt wird, ergibt sich hier ein Bottom-up Ansatz. Es sind aber auch andere Ausprägungsformen, gerade bei der Planung einer neuen Realisierung möglich, bei dem alle Ebenen gleichzeitig modelliert und angepasst werden.

Zu diesen drei Ebenen kommt noch ein Funktionsmodell, dessen Funktionsweise und Struktur wir in diesem Kapitel noch genauer vorstellen.

4.3.3 Verzahnung von Softwareentwicklung und IT-Betrieb

Wie bereits beschrieben, ist die Verzahnung der Softwareentwicklung und des IT-Betriebs eines der grundlegenden Anliegen des Modells. Um dies zu erreichen, müssen die Informationen, die bei der Entwicklung einer Anwendung entstehen und der Source-Code der Anwendung selbst möglichst weitergegeben und wieder verwendet werden.

Viele Informationen über die Abhängigkeiten auf der Ebene der Software lassen sich direkt aus dem Source-Code einer Anwendung ableiten. Diese Informationen können dann in das Softwaremodell integriert werden. Wir beschreiben in Kapitel 9.2.1 einen Prototyp, der Source-Code von Webanwendungen verwendet und daraus ein Modell der Softwarestruktur erstellt, das es dem Administrator ermöglicht, Regeln für eine Firewall automatisch zu generieren.

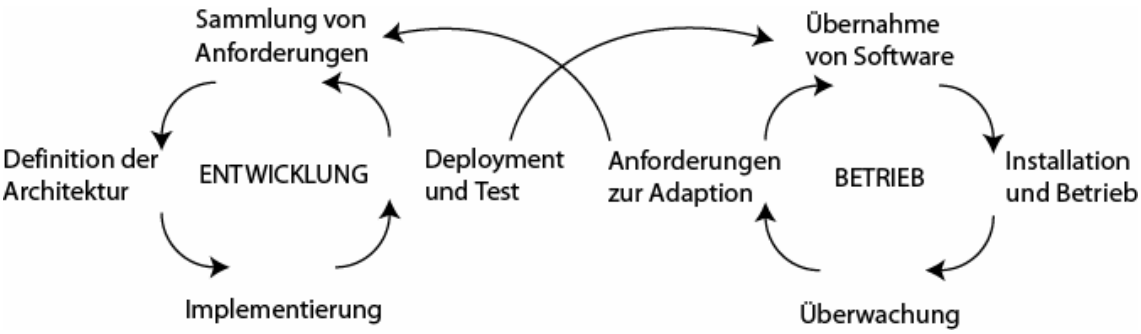

Abbildung 30 – Softwareentwicklung und Betrieb verzahnt

Wichtig ist jedoch, dass nicht nur eine Integration vom Entwickler zum Administrator, sondern auch in die umgekehrte Richtung erfolgt. Ein Beispiel hierfür ist die Pflege einer Anwendungslandschaft über einen längeren Zeitraum. Werden Basiskomponenten, die ein Anwendungssystem verwendet geändert, so muss der Administrator in der Lage sein, die Abhängigkeit der Anwendung von dieser Basiskomponente zu erkennen, festzustellen wie die Anwendung durch wen entsprechend abzuändern ist und welche Veränderungen der Entwickler durchführen muss. Für den Entwickler ist dabei vor allem wichtig, welche Schnittstellen und welche Basiskomponenten sich verändern. Die Abbildung 30 verdeutlicht diese Beziehung. Nur durch eine Integration der beiden Richtungen können so Prozesse etabliert werden, die komplexe Anwendungslandschaften auch über eine lange Zeit wart- und anpassbar halten. Dies beugt vor allem der Entstehung von Systemen vor, die nicht mehr gepflegt werden können und bei denen die Infrastruktur in einem unsicheren Zustand belassen werden muss, um die Laufzeit der Anwendung sicherzustellen.

4.4 Basiskonzepte unseres IT-Dienstmodells

Wir haben im vorigen Abschnitt unser Vorgehensmodell beschrieben. In diesem Abschnitt beschreiben wir die wichtigsten Konzepte und Annahmen, auf denen unser IT-Dienstmodell aufbaut.

4.4.1 Metamodell

Unsere Teilmodelle beschreiben Modelle die zur Erstellung konkreter Instanzen verwendet werden. Wir geben dazu die Vorschriften und Funktionen zur Erstellung an. Darüber hinaus geben wir Funktionen zur Konsistenzsicherung an. Die Informatik spricht in diesem Fall von Metamodellen.

„Modelle, die beschreiben, wie Modelle gebaut werden, nennt man Metamodelle“

Definition 15 – Metamodell (Quelle Wikipedia)

Die beschriebenen Objekte in den Teilmodellen sind die minimalen Informationen, die die Teilmodelle für die Modellierung der Beziehungen und der Abläufe innerhalb der Ebenen benötigen. Damit erleichtern wir die direkte Integration bereits existierender Modelle auf den jeweiligen Ebenen, indem wir den notwendigen Informationsgehalt möglichst klein halten. Andere Modelle, die sich auf das Schema des Teilmodells der jeweiligen Ebene abbilden lassen, lassen sich so nahtlos in unser IT-Dienstmodell integrieren, indem das Teilmodell der Ebene ersetzt wird. Wir geben in Kapitel 6.4 ein Beispiel dafür an, wie dies am Beispiel von ARIS erfolgen kann.

Alle unsere Teilmodelle basieren auf der Graphentheorie und lassen sich deshalb sehr einfach auch grafisch darstellen. Das schon beschriebene und in der Informatik und Mathematik etablierte Modellierungsmittel erleichtert den Einstieg.

4.4.2 Komponenten und Teilmodelle

Das Modell stützt sich vor allem auf die Beschreibung der Abhängigkeiten der Objekte der verschiedenen Ebenen und der Abläufe auf den Ebenen ab. Im Gegensatz zu einem integrativen Ansatz wie ARIS, das alle Informationen und Ebenen in einem Modell abbildet, besteht unser Modell auf vier verschiedenen Teilmodellen.

> *Teilmodelle beschreiben eine Ebene eines IT-Dienstes, die für sich alleine existieren kann und eine Realisierungsschicht eines IT-Dienstes beschreibt. Innerhalb des Teilmodells existieren Abläufe, die keine Beziehungen zu Elementen außerhalb des Teilmodells besitzen. Beziehungen zwischen verschiedenen Teilmodellen erlauben die Modellierung der verschiedenen Ebenen eines IT-Dienstes. Die Teilmodelle zusammengenommen ergeben unser IT-Dienstmodell.*

Definition 16 – Teilmodelle

Die vier verschiedenen Teilmodelle sind durch folgende Modelle gegeben:

- Funktionsmodell, zur Strukturierung und Beschreibung der realisierten Features eines IT-Dienstes
- Geschäftsprozessmodell, zur Beschreibung des realisierten Geschäftsprozesses
- Softwaremodell, zur Beschreibung der eingesetzten Anwendungen und Softwarekomponenten
- Infrastrukturmodell, zur Beschreibung der IT-Infrastruktur mit den eingesetzten IT-Systemen und der Netzwerkinfrastruktur

Durch die Verwendung von Teilmodellen lassen sich die einzelnen Ebenen einfacher handhaben und existierende Modelle der einzelnen Ebenen integrieren. Das Modell wird nicht durch Spezialinformationen der jeweiligen Ebenen überladen, wie dies bei einem integrativen Modell der Fall wäre.

Die Modellierung der Beziehungen zwischen den Ebenen führt mit den Teilmodellen zu einem Gesamtmodell des IT-Dienstes und bildet somit die Klammer um die verschiedenen Ebenen. Wir stellen die verschiedenen Bestandteile der Teilmodelle der verschiedenen Ebenen in den weiteren Kapiteln vor.

Die Modellierung der Objekte der einzelnen Ebenen erfolgt als Black-Box. Das interne Verhalten und die internen Zustände werden nicht modelliert. So werden zum Beispiel die Eigenschaften eines Servers, wie die Anzahl der Speichermodule nicht modelliert[32]. Wir lassen aber eine Verfeinerung der einzelnen Ebenen in Hierarchien zu. Dadurch wird das Konzept der Verfeinerung in die Teilmodelle integriert. Dies trägt vor allem zur Handhabbarkeit der verschiedenen Modelle bei.

Eine IT-Infrastruktur eines Unternehmens wird meist aus mehreren IT-Diensten bestehen. Selbst wenn IT-Dienste in Silos ausgeführt werden, wie schon eingeführt, besteht die IT-Infrastruktur aus vielen dieser Silos, die untereinander in Beziehung stehen. Durch die Integrierbarkeit existierender und unterschiedlicher Teilmodelle auf den verschiedenen Ebenen, kann unser Modell in einem solchen Fall auch zur Integration verschiedener Modellierungstechniken

[32] Eine Ausnahme könnte ein Server sein, dessen Speichermodule verschiedenen Partitionen einer virtuellen Maschine zugeordnet werden und bei dem auch die virtuellen Maschinen sehr detailliert modelliert werden sollen. In diesem Fall ließe sich sogar die Auswirkung des Ausfalls des Speichermoduls beschreiben.

auf den jeweiligen Ebenen eingesetzt werden, die so zu einem einheitlichen Modell verbunden werden.

4.4.3 Abhängigkeiten und Beziehungen

Das Modell dient der Abbildung von Abhängigkeiten innerhalb der Modellebenen und der Beschreibung der Beziehungen zwischen den Modellen. Dabei sind horizontale und vertikale Abhängigkeiten zu unterscheiden.

- Horizontale Abhängigkeiten - Abläufe innerhalb der Teilmodelle

  Beziehungen innerhalb der Teilmodelle sind Abläufe in Geschäftsprozessen, Aufrufbeziehungen zwischen Softwarekomponenten oder Kommunikationsbeziehungen zwischen IT-Systemen. Diese Beziehungen sind notwendig, um die Funktion der jeweiligen Ebene aufrecht zu erhalten.

  Für die Modellierung dieser Abhängigkeiten finden sich auf den verschiedenen Ebenen auch andere Beschreibungsmittel, so zum Beispiel die Prozessketten auf der Ebene der Geschäftsprozesse in ARIS. Diese Beziehungen sind für einen korrekten Ablauf auf der jeweiligen Ebene notwendig. Sind diese Beziehungen unterbrochen, können die entsprechenden Prozesse oder Komponenten nicht mehr korrekt ausgeführt werden. Wir sprechen deshalb in diesem Fall von Abhängigkeiten.

  Wir betrachten in diesem Fall nicht erst das Ereignis, dass ein Prozess wegen einer unterbrochenen Abhängigkeit nicht ausgeführt werden kann als Fehler, sondern bereits die Unterbrechung selbst. Wir nehmen dabei eine pessimistische Haltung ein, da eine Unterbrechung eventuell keine tatsächliche Auswirkung haben muss.

- Vertikale Abhängigkeiten: Beziehungen zwischen den Ebenen

  Die Verknüpfung zwischen den Teilmodellen setzt die jeweiligen Objekte und Komponenten der verschiedenen Ebenen zueinander in Beziehung. Diese Beziehung ordnet die Objekte einander zu, hier werden die Verbindungen zwischen Geschäftsprozessen, Softwarekomponenten, IT-Komponenten und Features des Funktionsmodells abgebildet.

  Diese Abhängigkeit ist im Gegensatz zur Abhängigkeit in den Ebenen aber nicht für die korrekte Funktion notwendig. Wir sprechen deshalb in diesem Fall nicht von einer Abhängigkeit, sondern von einer Beziehung.

Zur genauen Unterscheidung formalisieren wir die beiden Begriffe. Beziehungen beschreiben die Abbildungen zwischen den Ebenen.

> *Beziehungen beschreiben die Abbildung von Elementen unterschiedlicher Teilmodelle aufeinander. Jede Beziehung hat dabei mindestens zwei Elemente, die nicht zum selben Teilmodell gehören. Beziehungen sind für konkrete Abläufe innerhalb der Ebenen nicht notwendig sondern dienen der vollständigen Modellierung aller Ebenen eines IT-Dienstes.*

Definition 17 – Beziehungen

Betrachten wir zwei verschiedene Teilmodelle und betrachten nur die Beziehungen zwischen den Teilmodellen, so ergibt sich mindestens ein bipartiter, ungerichteter Graph, der die Beziehungen zwischen den beiden Ebenen beschreibt. Der Graph ist bipartit, da eine Beziehung immer mindestens ein Element aus je einem Teilmodell beinhaltet und wir keine Beziehungen zwischen Elementen eines Teilmodells zulassen[33]. Da die Abbildung der beiden Ebenen nicht vollständig sein muss, können sich mehrer Graphen ergeben. Die Abbildung der Ebenen erfolgt bidirektional, die Graphen beinhalten also keine Richtung.

[33] Dieses Modellelement wäre eine Abhängigkeit und keine Beziehung

Die Abbildung 31 verdeutlicht dies an den Elementen B und 2, die jeweils zu einander anderen Ebene gehören, zueinander aber in Beziehung stehen.

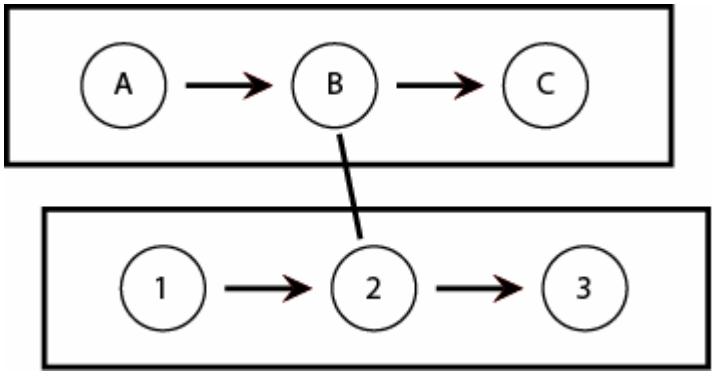

Abbildung 31 – Abhängigkeiten und Beziehungen

Abhängigkeiten beschreiben die Abläufe in den Ebenen:

> *Abhängigkeiten beschreiben Abläufe oder Aufrufbeziehungen innerhalb eines Teilmodells, die für die korrekte Ausführung des Teilmodells notwendig sind. Beispiel sind Aufrufbeziehungen von Softwarekomponenten, physikalische Netzwerkverbindungen oder Abläufe in Geschäftsprozessen.*

Definition 18 – Abhängigkeiten

Auch die Modellierung der Abhängigkeiten ergibt einen Graphen. Die Abbildung 31 verdeutlicht diesen innerhalb der Teilmodelle. Zuerst wird der Schritt A, dann der Schritt B und dann der Schritt C ausgeführt, bzw. der Schritt 3 nach dem Schritt 2, dieser nach dem Schritt 1. Im Gegensatz zu den Graphen der Beziehung sind die Graphen jedoch gerichtet. Die Richtung der Kanten wird beschrieben durch die Aufrufbeziehung[34].

Die Abbildung 32 verdeutlich nochmals den Unterschied zwischen Beziehungen zwischen den Teilmodellen und Abhängigkeiten innerhalb der jeweiligen Teilmodelle.

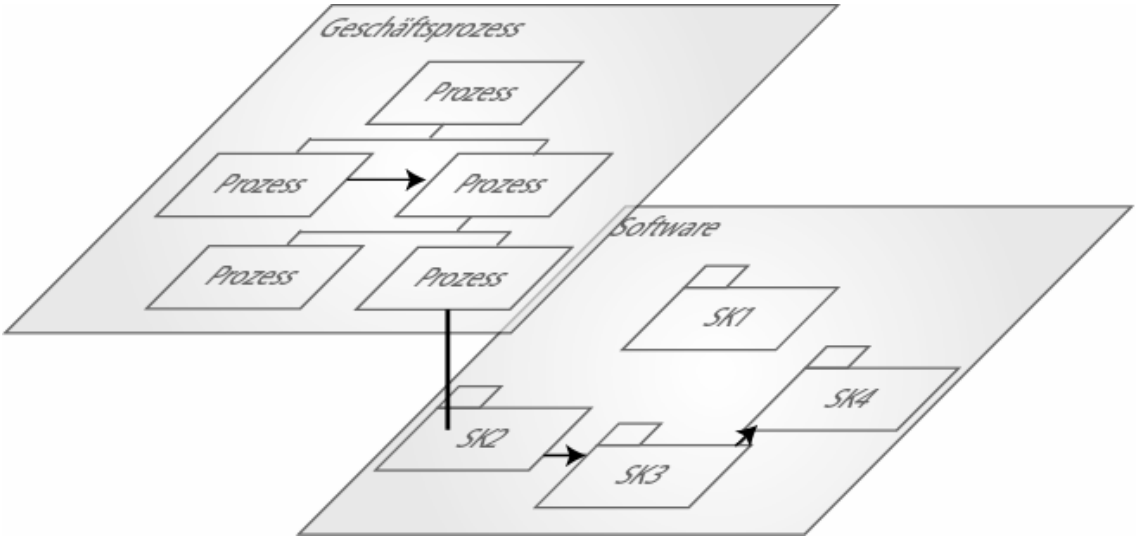

Abbildung 32 – Vertikale Beziehungen und Horizontale Abhängigkeiten

[34] Eine Besonderheit ist die Netzwerkinfrastruktur im Infrastrukturmodell, die ungerichtet ist

4.4.4 Verbindung über Schnittstellen und Kanalhierarchien

Die reale Kommunikation zwischen den Softwarekomponenten, bzw. zwischen den IT-Systemen erfolgt über definierte Schnittstellen auf verschiedenen Ebenen. Dazu gehören zum einen die durch die Softwarekomponenten bereitgestellten Schnittstellen, zum anderen die physikalische Netzwerkinfrastruktur.

Wir modellieren diese physikalischen Kanäle und auf Softwarekomponenten basierenden Schnittstellen durch Schnittstellen in unseren Teilmodellen. Die Schnittstellen existieren dabei sowohl im Softwaremodell, als auch im Infrastrukturmodell, können in Hierarchien angeordnet werden und modellieren so Schnittstellenhierarchien.

Bei der Übertragung von Informationen in Netzwerken werden verschiedene Ebenen eingesetzt, die ihre jeweiligen Aufgaben haben. Die bekanntesten Definitionen für Schnittstellenhierarchien sind der ISO-OSI Stack sowie die TCP/IP Protokollfamilie. Die verschiedenen Ebenen oder Schichten der Übertragung haben dabei unterschiedliche Aufgaben. Jede Schicht in einer Hierarchie ist dabei auf die Funktion der Schichten unterhalb angewiesen, um erfolgreich Daten übertragen zu können.

Die Abbildung 33 verdeutlicht dies an einem vereinfachten Beispiel. Hier sind zwei Server über ein Ethernetkabel miteinander verbunden. Diese Ethernetverbindung stellt die erste Ebene der Kanalhierarchie dar und wird durch die Netzwerkkarten der beiden Server sowie Funktionen des Betriebsystems zusammen mit den Netzwerkkartentreibern realisiert. TCP/IP setzt auf dieser Ethernetverbindung auf[35] und ermöglicht die Kommunikation zwischen den beiden Servern für Anwendungen über dieses Protokoll. In diesem Beispiel kommuniziert ein Webbrowser mit einem Webserver. Diese Kommunikation nutzt dabei die darunter liegenden Schichten. Auch die Kanäle bilden deshalb eine Hierarchie, um die Abhängigkeit der Kanäle voneinander zu modellieren.

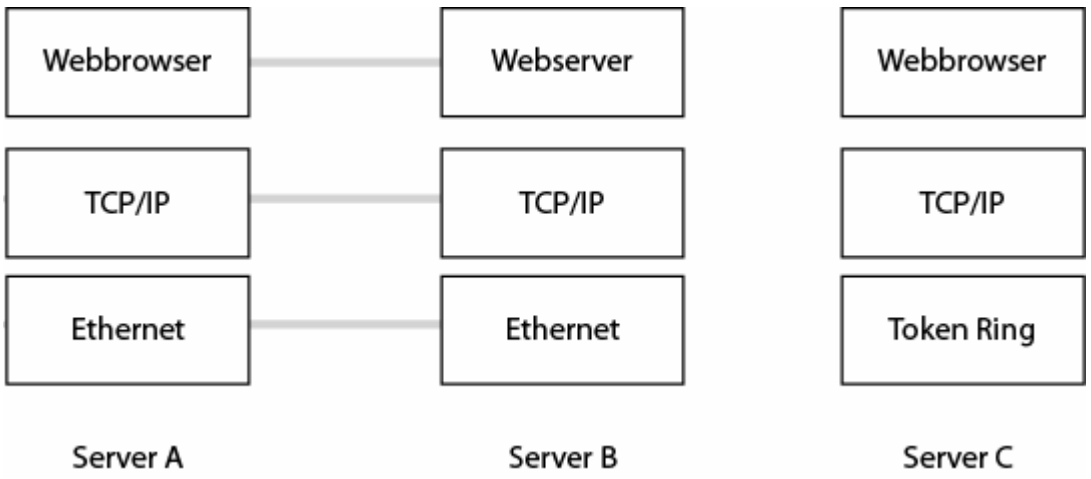

Abbildung 33 – Kanalhierarchien

Auf der Ebene der Software beschreiben die Schnittstellen die Aufrufbeziehungen zwischen den Softwarekomponenten. Die Verbindung ist gegeben durch eine Softwarekomponente, die eine Schnittstelle anbietet (im Beispiel der Webserver) und einer oder mehreren Softwarekomponenten, die die Schnittstelle benutzen (im Beispiel der Webbrowser). Diese Beziehung wird in Attributen der jeweiligen Komponenten modelliert.

Auf der physikalischen Ebene sind die Schnittstellenhierarchien an Endpunkte gebunden. Endpunkte modellieren Netzwerkkarten und die entsprechende Verkabelung in der Infrastruktur.

[35] Das Beispiel ist sehr vereinfacht dargestellt, auch TCP/IP ist in sich ein Protokoll mit Hierarchien

Zusammen mit den Kanälen und den Schnittstellen ergibt sich so die physikalische und logische Kommunikationsinfrastruktur einer IT-Infrastruktur. Die Möglichkeit der Modellierung der Schnittstellenhierarchien und der Kanalhierarchien ist eine Spezialität unseres Modells und ermöglicht so eine sehr feingranulare Modellierung der Kommunikationsbeziehungen, die weit über die Möglichkeiten von ARIS hinausgeht. Darüber hinaus ist durch die Modellierung der Kanalhierarchien auch eine Möglichkeit gegeben, virtuelle Kanäle auf physikalische Kanäle abzubilden und so die Containerbeziehungen der virtuellen Verbindungen auf reale Verbindungen darzustellen. Damit ist zum Beispiel auch der Einfluss der Veränderung von Firewalleinstellungen auf die Kanäle und Verbindungen für Softwarekomponenten analysierbar.

Durch die Abstraktion der verschiedenen Ebenen und die Modellierung als Schnittstellenhierarchie werden die Feinheiten der verschiedenen Realisierungen abstrahiert. Verschiedene Netzwerkstrukturen und Topologien werden dadurch sehr einfach dargestellt und auch für einen technisch nicht versierten Anwender verständlich. Im Beispiel wird dies durch den Server C verdeutlicht. Im Gegensatz zu Server A und B hat dieser Server eine Token-Ring Netzwerkkarte, ein zu Ethernet inkompatibles Protokoll. Die beiden Schnittstellen von Server B und Server C passen deswegen nicht zusammen und die Kanalhierarchie mit den höheren Ebenen kann nicht modelliert werden, es existieren keine Kanäle zwischen Server C und den Servern A und B, da die Server schon auf der untersten Ebene nicht miteinander kommunizieren können[36]. An den Endpunkt von Server C ist entsprechend eine Token-Ring Karte modelliert, was sich in der Schnittstellenhierarchie entsprechend Token-Ring widerspiegelt. Durch diese Art der Modellierung wird die technische Tatsache der unterschiedlichen Netzwerkinfrastruktur in der Schnittstellenhierarchie sehr einfach und auch für den Laien deutlich. Diese Modellierung ist deutlich einfacher, als die Verwendung von unterschiedlichen Klassen und Objekten für Endpunkte in Modellen und Schemata wie CIM.

4.4.5 Autonomie und Teilautonomie

Wie bereits eingeführt, existieren innerhalb der Teilmodelle Abhängigkeiten zwischen den Objekten. Dabei müssen jedoch nicht alle Elemente eines Teilmodells miteinander interagieren. Innerhalb der verschiedenen Teilmodelle existieren in den meisten Fällen Abläufe, die völlig autonom voneinander sind. Diese unabhängigen Abläufe bilden im Sinne der Graphentheorie einzelne Graphen, die unabhängig voneinander existieren.

4.4.6 Verschattung / Verfeinerung

Ein wichtiges Mittel der Modellierung ist die Verfeinerung, um einzelne Aspekte eines Modells detaillierter darzustellen oder ein Modell während der Modellierung um zusätzliche, detailliertere Informationen anzureichern. Die Verschattung ist der entgegengesetzte Vorgang der von mehreren verschiedenen Objekten auf ein einziges Objekt abstrahiert. Bei der Verfeinerung werden Informationen hinzugefügt, bei der Verschattung gehen Informationen verloren.

Abbildung 34 – Verfeinerung und Verschattung

[36] Es gibt Netzwerkkomponenten, die zwischen Token-Ring Topologien und Ethernet vermitteln können. Existiert eine entsprechende Komponente, würde diese zwischen den beiden Servern modelliert. Die in der Hierarchie höher angeordneten Kanäle werden dann auch den Kanal zwischen den Servern und dieser Komponente abgebildet.

Die Abbildung 34 beschreibt die Verfeinerung und Verschattung am Beispiel eines Server. Bei der Verfeinerung wird eine Festplatte als Bestandteil des Servers modelliert. Bei der Verschattung wird diese Information aus dem Modell entfernt.

Wir definieren für unsere Teilmodelle die entsprechenden Funktionen. Diese sind vor allem deshalb wichtig, um die Konsistenz der jeweiligen Teilmodelle zu sichern.

Eine Spielart der Verfeinerung und Verschattung sind Zoomfunktionen in Anwendungen. Bei diesen wird nur die Darstellung verändert und die Modelle enthalten auch bei der Verschattung weiterhin die Informationen, diese werden nur nicht mehr dargestellt.

4.4.7 Modellierung von Aktionen / Reaktionen auf abstrakter Ebene

Unser Modell soll die Auswirkungen von Ausfällen und Veränderungen an den Bestandteilen eines IT-Dienstes modellieren. Dazu werden an de einzelnen Komponenten des IT-Dienstes fail() und recover() Funktionen modellieren, die das Verhalten bei einem Ausfall beschreiben.

Diese Funktionen sind abstrakt und könnten zum Beispiel in einem Tool ausgeführt werden. Wir definieren für diese Funktionen keine Sprache, sondern überlassen diese Entscheidung einem Toolentwickler, der das Modell umsetzt.

Für diese Arbeit verwenden wir zur Modellierung der Funktionen Pseudo-Code, der sich an einer Java/C# Syntax orientiert.

4.4.8 Instanzen und Eindeutigkeit

Häufig kommt es vor, dass Softwarekomponenten und Anwendungssysteme mehrmals in der Infrastruktur installiert sind und damit die Schnittstellen auf einer Ebene mehrfach vorhanden sind. Eindeutig werden die Softwarekomponenten durch die Bindung an ein IT-System. Soll bereits davor eine Bindung zwischen einem Geschäftsprozess und einer Softwarekomponenten modelliert werden, macht diese Beziehung die Softwarekomponente eindeutig. Wird die Softwarekomponente erneut modelliert, so existiert hier die Beziehung zum Geschäftsprozess nicht.

Dies entspricht auch der technischen Realisierung. Ein Port auf der Ebene von TCP kann nur von einer Softwarekomponente auf einem IT-System an einem Endpunkt gebunden werden. Die Verbindung zwischen Softwarekomponenten bei der Existenz verschiedenen Instanzen innerhalb der Teilmodelle wird durch die Modellierung von Kanälen eindeutig. Damit wird in unserem Beispiel aus Webbrowser und Webserver eindeutig festgelegt, mit welchem Webserver auf welchem physikalischen Server der Webbrowser interagiert.

Im Funktionsmodell sind Features in eine Hierarchie eingeordnet und ihr Bezeichner ist eindeutig. Auch IT-Systeme werden durch ihre Endpunkte eindeutig innerhalb des Infrastrukturmodells, da durch die Endpunkte jedem IT-System ein eindeutiger Bezeichner durch eine eindeutige Schnittstelle zugeordnet wird[37].

Ein Sonderfall sind Clustersysteme und Hochverfügbarkeitslösungen, auf die wir in Kapitel 5.1.5 genauer eingehen. Hier realisieren mehrere physikalische IT-Systeme ein virtuelles IT-System.

4.5 Modellebenen

Wir haben seither Grundkonzepte beschrieben die für alle Teilmodelle gelten. Wir stellen in diesem Abschnitt die verschiedenen Teilmodelle, ihre Funktionen und Komponenten vor und gehen auf den jeweiligen Umfang der Teilmodelle ein.

Die verschiedenen Teilmodelle sind über Beziehungen miteinander verbunden. Sie beschreiben Geschäftsprozesse, Software, die IT-Infrastruktur und das Funktionsmodell. Alle Teilmodelle beschreiben die Abläufe und Abhängigkeiten auf den jeweiligen Ebenen. Mit diesen vier Teil-

[37] Dies entspricht auf der technischen Ebene dem Konzept, dass jede Netzwerkkarte eine weltweit eindeutige MAC/WWNID besitzt

modellen und den Beziehungen ergibt sich unser IT-Dienstmodell, das den IT-Dienst beschreibt.

4.5.1 Geschäftsprozessmodell

Das Geschäftsprozessmodell beschreibt die Geschäftsprozesse, die ineinander verschachtelt werden können und die Abhängigkeiten der einzelnen Geschäftsprozesse voneinander. Die Abbildung 35 verdeutlicht dies an einem Beispiel, das wir in diesem Abschnitt schrittweise erweitern werden. Das Beispiel beschreibt einen sehr einfachen Webshop, bei dem Kunden Waren bestellen können. Alle weiterhin notwendigen Schritte werden zur Vereinfachung nicht modelliert, wir konzentrieren uns auf das Anlegen von Bestellungen.

Der Geschäftsprozess Bestellung unterteilt sich in die beiden untergeordneten Geschäftsprozesse Kundendaten erfassen und Bestellung erfassen. Der Geschäftsprozess Bestellung erfassen ist dabei ein nachgelagerter Geschäftsprozess des Geschäftsprozesses Kundendaten erfassen und damit von diesem abhängig. Diese Abhängigkeit wird dargestellt durch die gerichtete Kante zwischen den beiden Geschäftsprozessen.

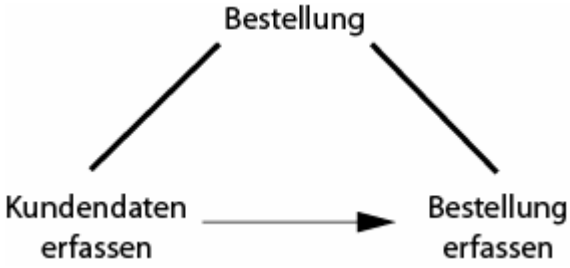

Abbildung 35 – Geschäftsprozess mit zwei untergeordneten Geschäftsprozessen

Über die bereits beschriebenen Beziehungen werden die Geschäftsprozesse auf die Softwarekomponenten bzw. die Features des Funktionsmodells abgebildet. Die Modellierung der Beziehung zur IT-Infrastruktur erfolgt indirekt, durch die Modellierung der Verteilung der Softwarekomponenten in der IT-Infrastruktur. Daraus ergeben sich dann die Beziehungen zwischen IT-Systemen und Geschäftsprozessen. Damit lassen sich Ausfälle auf der IT-Infrastrukturebene auf die Ebene der Geschäftsprozesse abbilden und die betroffenen Prozesse identifizieren. Werden auf der Geschäftsprozessebene auch Informationen über die Wertschöpfung einzelner Geschäftsprozesse hinterlegt, so lassen sich entsprechende Ausfälle auch monetär bewerten[38].

Gerade das Funktionsmodell erleichtert zum einen die Kommunikation des Geschäftsprozessverantwortlichen mit dem Entwickler, da die Funktionen eines Geschäftsprozesses auf ein Feature des Funktionsmodells abgebildet werden, auf das auch die Softwarekomponente abgebildet ist, die das Feature realisiert. Dadurch wird die Beziehung an die Funktion gebunden und somit eine Bedeutung beschrieben. Zum anderen lassen sich so aber auch verschiedene Geschäftsprozesse vergleichen, indem diese Features verwendet werden, um unterschiedliche Realisierungen zu vergleichen, oder aber Geschäftsprozesse und die Verwendung von Softwarekomponenten zu vereinheitlichen und so zur Standardisierung der Softwarelandschaft beizutragen.

[38] Das Modell bildet Abhängigkeiten auf der Basis von Nachrichtenaustausch ab. Ist eine Abhängigkeit nicht erfüllt, so ergibt sich anhand des Modells ein Fehler. Ob dieser Fehler schon zum Tragen kommt, weil es sich um ein System mit ständigem Nachrichtenaustausch handelt, oder ob der Fehler erst in 2 Stunden eintritt, weil erst dann wieder Daten übertragen werden, bildet unser Modell nicht direkt ab. Dahingehend kann eine monetäre Bewertung an dieser Stelle bei asynchronen Prozessen schwierig sein, sie ist aber im Rahmen der Modellierung der entsprechenden Funktion möglich. Diese Monetäre Bewertung ist aber ein eigenes Forschungsgebiet.

Für einen Geschäftsprozessverantwortlichen ist es mit unserem Modell möglich, seine Geschäftsprozesse zu modellieren oder abzubilden und ihre Abhängigkeiten von der Software als auch der IT-Infrastruktur zu analysieren. Darüber hinaus werden Abhängigkeiten zwischen Geschäftsprozessen sichtbar, die sich möglicherweise erst durch die verwendete Software ergeben. Fehler in anderen Ebenen werden für ihn mit ihren Auswirkungen auf die Geschäftsprozesse erkennbar.

Bei der Planung neuer Geschäftsprozesse, sowie Änderungen an Geschäftsprozessen, erleichtert ihm das Modell die Analyse der betroffenen Systeme und Komponenten auf anderen Schichten. So kann zum Beispiel die Frage beantwortet werden, welche IT-Systeme bei der Auslagerung eines Geschäftsprozesses im Rahmen eines Outsourcingprojektes betroffen sind, ob dabei Ressourcen eventuell nicht mehr benötigt werden bzw. ob dadurch auch andere Geschäftsprozesse ausgelagert werden müssen, bzw. die Kontrolle über wichtige Datensätze verloren geht.

Darüber hinaus ist es möglich, Servicelevelagreements auf der Ebene der Geschäftsprozesse auf die Ebenen der Software und der IT-Infrastruktur abzubilden und damit verbundene Anforderungen entsprechend zu dokumentieren und zu veranschaulichen.

Unser Modell ermöglicht die Beschreibung von Abhängigkeiten zwischen Geschäftsprozessen und der für den IT-Betrieb notwendigen Informationen. Es ist jedoch kein Mittel zur Geschäftsprozessmodellierung. Es fehlt zum Beispiel die Möglichkeit der Modellierung von Aktionen der einzelnen Geschäftsprozesse, auf die wir bewusst verzichten, um das Modell für seinen Einsatzzweck kompakt zu halten. Für die detaillierte Modellierung von Geschäftsprozessen sollte auf andere Mittel wie ARIS oder das Konzept von Thurner [Thu04] zurückgegriffen werden. In einem späteren Kapitel geben wir eine Möglichkeit zur Integration von ARIS Modellen an.

4.5.2 Funktionsmodell

Das Funktionsmodell dient zur besseren Strukturierung von Modellen. Im Funktionsmodell werden Features hinterlegt, die die Funktionen beschreiben, die Elemente auf den verschiedenen Ebenen ausführen. Das Funktionsmodell beschreibt im Gegensatz zu den anderen Modellen nicht den konkreten Ablauf oder die konkrete Implementierung, sondern es abstrahiert die jeweiligen Elemente und strukturiert diese nach der jeweiligen Funktion der Objekte.

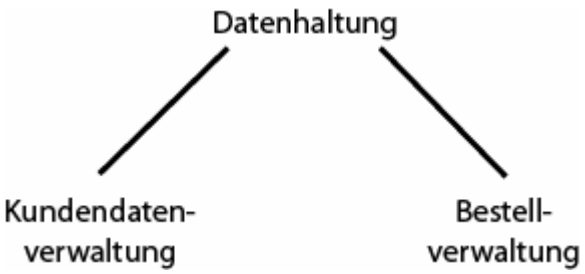

Abbildung 36 – Funktionsmodell mit Features

Die Abbildung 36 beschreibt das Funktionsmodell für unser Beispiel. Das Feature Datenhaltung wird dabei unterteilt in die beiden Teilfeatures Kundendatenverwaltung und Bestellverwaltung. Die Zuordnung der Geschäftsprozesse verdeutlicht Abbildung 37. Dabei werden die Geschäftsprozesse auf die jeweiligen Features abgebildet, die sie benutzen.

Teile dieser Informationen des Funktionsmodells finden sich auch indirekt in den Beziehungen zwischen den Teilmodellebenen. Dabei wird zum Beispiel der Schritt „Bestellung erfassen“ aus einem Geschäftsprozess auf eine Softwarekomponente zugewiesen. Auf der Funktionsebene wird dieses Feature „Bestellverwaltung“ modelliert, die Beziehung findet sich aber auch in der Beziehung zwischen der Softwarekomponente und dem Geschäftsprozess.

Auf den ersten Blick beschreibt das Funktionsmodell hier nur die Beziehung zwischen den beiden Modellelementen erneut, dies könnte auch durch eine Bezeichnung der Beziehung zwischen den beiden Ebenen erreicht werden. Werden aber mehrere Geschäftsprozesse mit derselben Funktionalität modelliert, die durch unterschiedliche Softwarekomponenten realisiert werden, so werden diese gleichartigen Funktionsbeziehungen nicht mehr deutlich. Dies gilt auch für den Umkehrschluss. Die Beziehungen zwischen den Softwarekomponenten und den Geschäftsprozessen werden durch die Abbildung auf das Funktionsmodell nicht überflüssig, da mehrere Softwarekomponenten dieselben Features des Funktionsmodells realisieren können. Erst durch die Abbildung zwischen den Geschäftsprozessen und den Softwarekomponenten wird die Wahl der Implementierung beschrieben und im Modell hinterlegt.

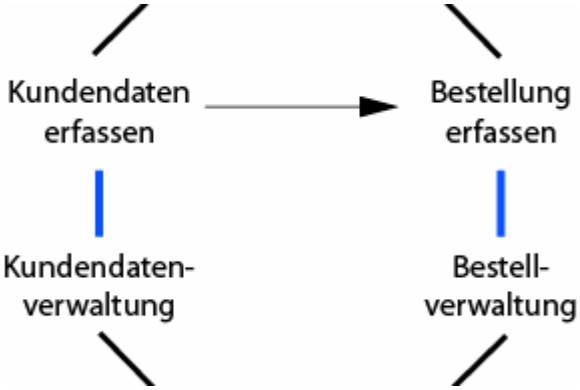

Abbildung 37 – Geschäftsprozesse und Features

Das Funktionsmodell erlaubt, wenn mehrere Varianten zur Realisierung zur Verfügung stehen, also auf ein Feature mehrere Elemente einer anderen Ebene zugeordnet werden, die Beantwortung der Frage, welche verschiedenen Realisierungen dieses Feature umsetzen. Darüber hinaus lassen sich so verschiedene Realisierungen vergleichen. Im Falle von Softwaresystemen ermöglicht dies zum Beispiel, die Eignung eines Softwaresystems für eine Realisierung eines Geschäftsprozesses zu überprüfen und verschiedene Softwaresysteme so durch den Vergleich der Abdeckung der geforderten Features zu vergleichen. Gleiches gilt für Infrastrukturmodelle und Geschäftsprozessmodelle.

4.5.3 Softwaremodell

Das Softwaremodell beschreibt die Anwendungssysteme, die Softwarekomponenten, aus denen diese bestehen, sowie die Abhängigkeiten zwischen diesen Systemen. Für den Softwareentwickler wird es dadurch möglich, Abhängigkeiten zu beschreiben und verschiedene mögliche Varianten zu dokumentieren, wie seine Software auf der IT-Infrastruktur ausgeführt werden kann. Diese Referenzarchitekturen sind heute noch meist textuell beschrieben und ihre Darstellung nicht normiert. Durch unser Modell können diese Bausteine sehr leicht auf eine konkrete Infrastruktur abgebildet werden. Dadurch entsteht ein durchgängig modellbasierter Prozess, der Fehler durch die Vermeidung von Medienbrüchen reduziert und die Kommunikation zwischen Entwicklung und Betrieb einer Applikation verbessert.

Die Abbildung 38 beschreibt wieder unser Beispiel. Dieses Modell beinhaltet noch keine Softwarekomponenten, um das Beispiel nicht zu überfrachten. Das Softwaremodell beinhaltet drei Anwendungssysteme, ein CRM System für das Customer Relationship Management, also die Kundenverwaltung und einen Webserver, auf dem der Webshop betrieben wird. Beide Systeme stützen sich auf einer Datenbank ab, die die Daten der beiden Systeme ablegt. Die Abhängigkeit ist wieder durch gerichtete Kanten verdeutlicht. Eine vereinfachte Modellierung der Schnittstellen lautet:

*Datenbankschnittstelle: * : SQL*

Entsprechend haben die beiden Anwendungssysteme ein Requirement-Attribut, das die Abhängigkeit von der durch die Datenbank bereitgestellte Schnittstelle beschreibt.

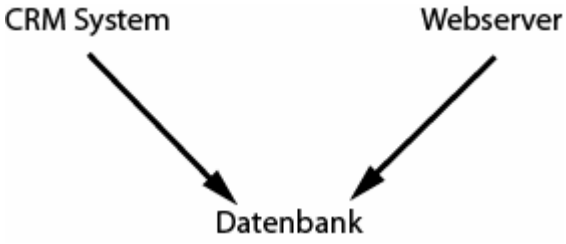

Abbildung 38 – Anwendungen mit Abhängigkeiten

*CRM.req = * : SQL*

*Webserver.req = * : SQL*

Die Softwarekomponenten werden auf das Funktionsmodel und die jeweiligen Features abgebildet. Abbildung 39 verdeutlicht dies für unser Beispiel. Die Datenbank wird auf das Feature Datenhaltung abgebildet, das CRM System auf die Kundendatenverwaltung und der Webserver auf die Bestellverwaltung[39].

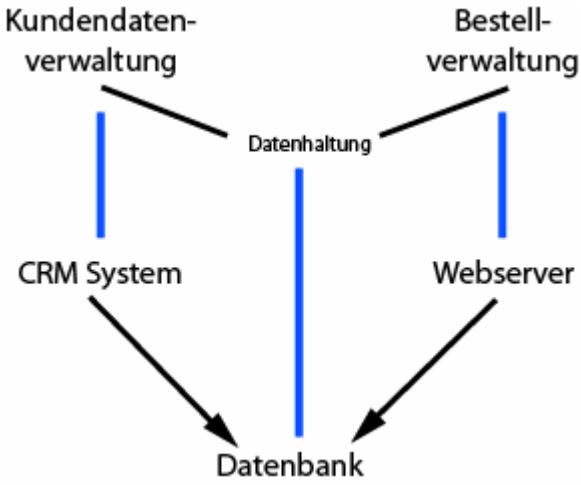

Abbildung 39 – Anwendungssysteme und Features

Diese Abbildung auf das Featuremodell helfen dem Geschäftsprozessverantwortlichen und dem Softwareentwickler, ein gemeinsames Modell des IT-Dienstes zu entwerfen, da die Beziehungen zwischen dem Softwaremodell und dem Geschäftsprozessmodell aufbauend auf den jeweiligen Beziehungen zu Features des Funktionsmodells benutzt werden können. Die Abbildung 40 verdeutlicht diese Beziehungen. Neben der Abbildung zwischen dem Softwaremodell und dem Geschäftsprozessmodell auf die jeweiligen Features sind hier auch die Beziehungen zwischen dem Geschäftsprozessmodell und dem Softwaremodell beschrieben. Diese ergeben sich durch die gemeinsamen Features.

Wie bereits im vorigen Abschnitt beschrieben, lässt sich darüber hinaus durch die Abbildung verschiedener Anwendungssysteme auf das Featuremodell eines Geschäftsprozesses die Entscheidung, welches Anwendungssystem die geforderten Features am besten abdeckt, einfacher beantworten. Die Abbildung auf die jeweiligen Features ermöglicht hier einen leichten Zugriff auf die unterschiedlichen Verfahren, die zum selben Zweck eingesetzt werden. Dies kann auch zur Konsolidierung beitragen. Lassen sich die entsprechenden Prozesse und Fea-

[39] Die Hierarchie ist zur Vereinfachung der Darstellung auf den Kopf gestellt

tures auch mit der Kaufsoftware B realisieren und ist diese auf der Infrastruktur ausführbar, so ist eine Migration möglich. Ähnlich lassen sich auch Migrationen und verschiedene Versionen der Infrastruktur bewerten. Werden alle heute vorhandenen Dienste und Softwarekomponenten, sowie die zukünftige Infrastruktur modelliert, so lässt sich die Frage der Ausführbarkeit und Vollständigkeit der neuen Infrastruktur durch eine Abbildung der Geschäftsprozesse und der entsprechenden Softwarekomponenten auf die neue Infrastruktur beantworten.

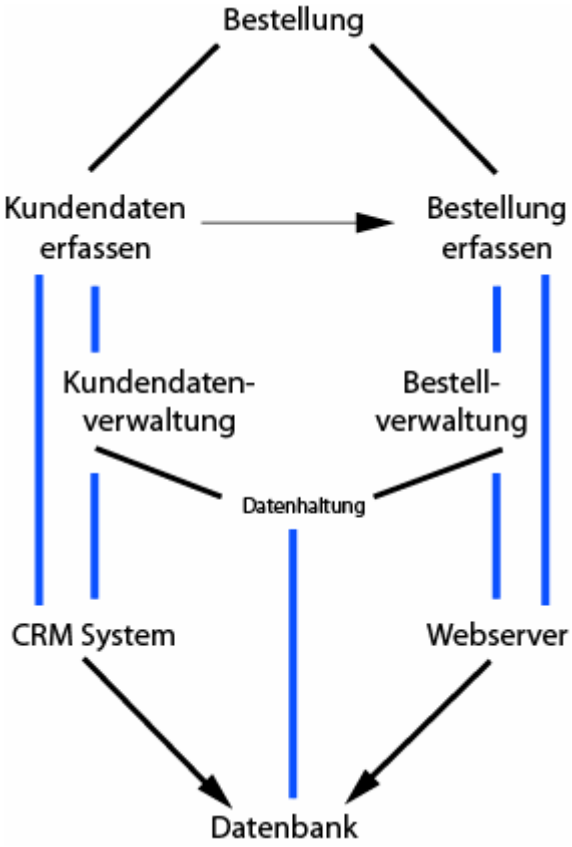

Abbildung 40 – Geschäftsprozesse, Anwendungssysteme und Features verknüpft

Unser Modell ermöglicht hier aber auch die Beschreibung von Standards in einem Unternehmen. So ist es möglich, einen Beispielgeschäftsprozess auf eine Standardsoftware im Unternehmen abzubilden, die nur als Platzhalter dient. Damit existiert eine konkrete Vorgabe, die nur instantiiert werden muss. Darüber hinaus kann so eine Mindestabdeckung von Features für Softwarekomponenten beschrieben werden.

4.5.4 Infrastrukturmodell

Auf der Ebene der Infrastruktur werden die IT-Systeme beschrieben und die Netzwerkinfrastruktur modelliert. Die Abbildung 41 verdeutlicht dies für unser Beispiel an einem sehr einfachen Szenario. Drei Server sind mit einer sehr einfachen Netzstruktur verbunden, es existiert jeweils ein Kanal zwischen dem CRM Server und dem Webserver mit dem Datenbankserver.

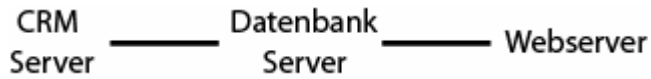

Abbildung 41 – Infrastrukturmodell mit drei Servern

Neben Servern werden im Infrastrukturmodell auch aktive Netzwerkkomponenten wie Switches, Hubs oder Firewalls modelliert, zur Vereinfachung verzichten wir darauf in unserem Beispiel. An die Server sind jeweils Endpunkte modelliert, die die jeweiligen Netzwerkkarten darstellen. Auch hier wurde zur Vereinfachung auf eine graphische Darstellung verzichtet.

Es ist mit dem Modell möglich, Engpässe wie „Single Points of failure“ in einer IT-Infrastruktur zu identifizieren. Lassen sich die Informationen und Komponenten eines Modells mit Informationen aus einem Monitoringsystem hinterlegen, die die Auslastung und das Verhalten der Systeme vergleichbar machen, so ist auch eine Aussage über die Qualität einer Infrastruktur möglich. Durch die Modellierung von Abhängigkeiten und das Wissen über die vorhandenen Beziehungen in der IT-Infrastruktur wird darüber hinaus die Planung bei der Einführung neuer Anwendungen deutlich vereinfacht. Ein IT-Verantwortlicher kann durch das Modell erkennen, welche Systeme von einer Neueinführung betroffen sind und ob sich deshalb eventuelle Probleme ergeben können.

An die Endpunkte der Server sind jeweils die bereitgestellten Schnittstellen modelliert. Die Endpunkte sind mit den jeweiligen Kanälen verbunden. In diesem Beispiel:

Endpunkt: Ethernet[CRM Server]

Endpunkt: Ethernet[SQLServer]

Endpunkt: Ethernet[Webserver]

Kanal A1 (Ethernet[CRM Server], Ethernet[Datenbank Server])

Kanal B1 (Ethernet[Webserver], Ethernet[Datenbank Server])

Die Abbildung zwischen den Softwarekomponenten und den IT-Systemen erfolgt bei der Installation dieser Systeme auf dem jeweiligen Server. Die Abbildung 42 verdeutlicht dies für unser Beispiel.

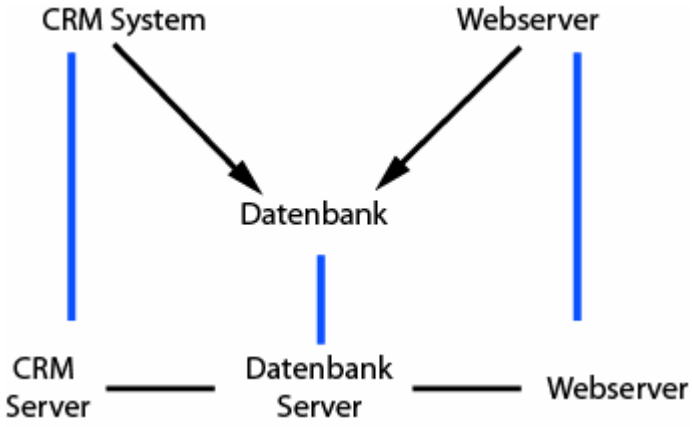

Abbildung 42 – Anwendungssysteme auf Servern

Bei der Einrichtung der Softwarekomponenten auf den IT-Systemen, werden auch die Schnittstellen and en Endpunkten der Server entsprechend angepasst und zusätzliche Kanäle, die die Softwarekomponenten benötigen, beschrieben.

Endpunkt: Ethernet[CRM Server]

Endpunkt: Ethernet[SQLServer] : SQL

Endpunkt: Ethernet[Webserver]

Datenbankschnittstelle: Ethernet[SQLServer] : SQL

Kanal A1 (Ethernet[CRM Server], Ethernet[Datenbank Server])

Kanal A2 (Ethernet[CRM Server] : SQL, Ethernet[Datenbank Server]: SQL)

Kanal B1 (Ethernet[Webserver], Ethernet[Datenbank Server])

Kanal B2 (Ethernet[Webserver] : SQL, Ethernet[Datenbank Server] : SQL)

Die Datenbankschnittstelle wird bei der Installation auf dem Datenbankserver entsprechend verfeinert. Die logischen Kanäle A2 und B2 beschreiben dabei keine Netzwerkinfrastruktur, sondern mit ihrem Schnittstellenstack die Beziehung zwischen dem CRM- bzw. dem Webserver und der Datenbank.

Die Abbildung 43 beschreibt den IT-Dienst unseres Beispiels mit allen Teilmodellen und den entsprechenden Beziehungen.

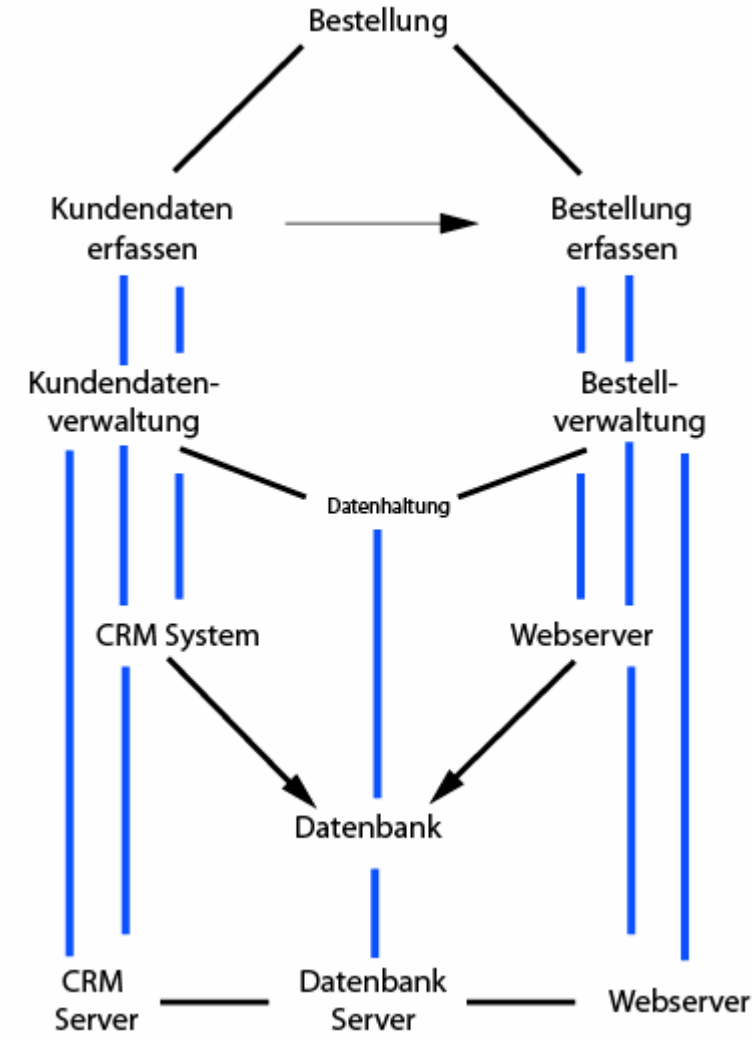

Abbildung 43 – Modell eines IT-Dienstes mit Teilmodellen

5 Modellelemente

Wir haben im vorigen Kapitel grundlegende Konzepte und Strukturen unseres Modells beschrieben. In diesem Kapitel definieren wir die Begriffe und Elemente der unterschiedlichen Teilmodelle und geben die Funktionen zur Erstellung eines IT-Dienstmodells und der verschiedenen Teilmodelle an. Das Modell spaltet sich in die vier Ebenen Geschäftsprozesse, Softwareebene, Infrastruktur und Funktionen (Abbildung 44).

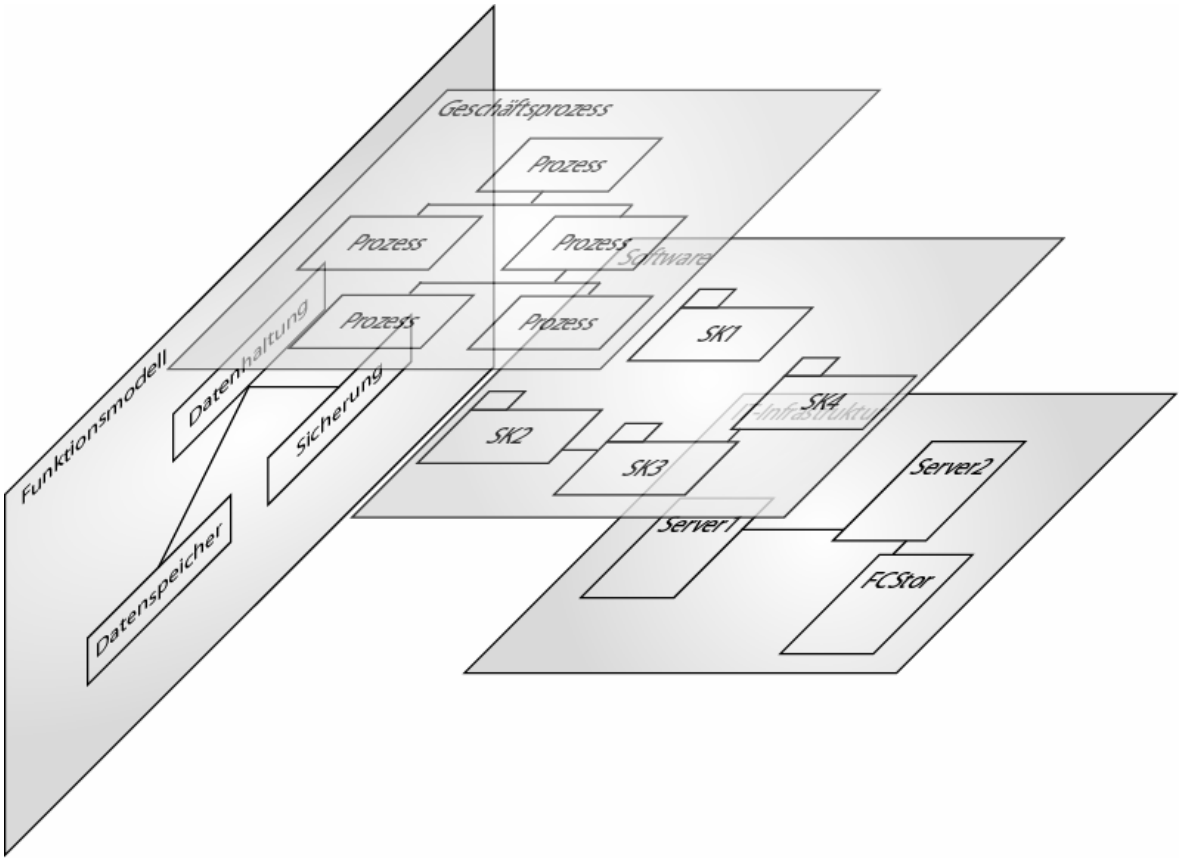

Abbildung 44 – Teilmodelle des Modells

Eines der Grundkonzepte unseres Modells ist die Abbildung verschiedener Teilmodelle aufeinander und damit die Modellierung der Beziehung dieser Modelle untereinander. Das Funktionsmodell wird unter anderem dazu verwendet, die Bedeutung und den Grund dieser Abbildung zu beschreiben. Werden die Beziehung zwischen dem Geschäftsprozessmodell und dem Softwaremodell abgebildet, so sind auch die Elemente der jeweiligen Ebene mit den gemeinsamen Features des Funktionsmodells verbunden. Beteiligt an diesem Schritt in der Modellerstellung sind der Geschäftsprozessverantwortliche und der Softwareentwickler, die ausgehend von eigenen Modellen oder einem Teilmodell nach unserer Definition gemeinsam ein Modell entwerfen.

Die Abbildung der Softwarekomponenten des Softwaremodells auf die IT-Infrastruktur, die im Infrastrukturmodell beschrieben wird, ergibt dann das Gesamtmodell. Dazu werden die Informationen des Entwicklers über die Softwarekomponenten vom Administrator verwendet und die Softwarekomponenten auf den IT-Systemen eingerichtet. Diese Installation kann durch einen Administrator oder eine Anwendung einmalig erfolgen, sie kann aber auch im Fall von Virtualisierungslösungen wie FSC Flexframe/ASCC/Bladeframe, dem SUN N1 Gridcontainer oder anderen Systemen automatisch und sehr dynamisch, je nach aktueller Last der Einzelsysteme

durchgeführt werden, eventuell sogar mehrmals täglich in eine Tag- und Nachtkonfiguration[40]. Zusammen mit den Beziehungen zwischen den Softwarekomponenten und dem Funktionsmodell werden auch die IT-Systeme den Features im Funktionsmodell zugeordnet. Durch die Modellierung der Beziehungen zwischen Softwarekomponenten und Geschäftsprozessen entsteht eine ganzheitliche Sicht auf einen IT-Dienst.

5.1 Übergreifende Elemente

Unser Modell umfasst vier Teilmodelle, deren Beziehungen zueinander modelliert werden. Dadurch entsteht ein Gesamtbild auf einen IT-Dienst. Wir definieren zuerst einige Elemente und Konzepte, die sich auf allen Teilebenen des Modells wieder finden. Diese sind teilweise Voraussetzung für die Funktion des Modells wie Endpunkte und Schnittstellen sowie das Konzept der Komponentenverfeinerung und dienen der Verknüpfung der Teilmodelle untereinander.

5.1.1 IT-Dienst

In der Literatur existieren sehr viele verschiedene Definitionen für einen Service. Die Definition eines IT-Services[41] oder eines IT-Dienstes im Sinne der Arbeit lautet:

> *Ein IT-Dienst ist die Erbringung eines oder mehrerer Geschäftsprozesse durch eine Softwarelandschaft, die auf einer IT-Infrastruktur ausgeführt wird. Bestandteil des IT-Dienstes sind die Geschäftsprozesse, die Softwarekomponenten, die verwendeten IT-Infrastrukturkomponenten sowie die Beschreibung ihrer Abhängigkeiten und gegenseitigen Beziehungen*

Definition 19 – IT-Dienst

Es ist sehr schwierig, die Grenzen eines Geschäftsprozesse genau anzugeben. Theoretisch lassen sich auch alle Geschäftsprozesse eines Unternehmens als ein großer Geschäftsprozess auffassen. Andererseits lässt sich fast jeder Geschäftsprozess nochmals unterteilen, indem Details hinzugefügt werden. Wir überlassen die Wahl einer für das Modell sinnvollen Granularität der Geschäftsprozess den jeweils Verantwortlichen.

Für einen IT-Dienst sind in einem Servicelevelagreement (SLA) die Parameter für dessen Ausführung spezifiziert. Diese Parameter umfassen unter anderem die Verfügbarkeit, die Fehlertoleranz, Wartungsfenster, etc.. Darüber hinaus kann ein IT-Dienst in seinem SLA eine Definition von anderen, nichttechnischen Dienstleistungen wie Support, Ausweichmöglichkeiten auf andere Rechenzentren oder technische Hilfestellungen beinhalten. Der IT-Dienst definiert sich über sein SLA und die Zeit, über die seine Erbringung vertraglich fixiert wurde und ist in unserem Modell ein nicht auf die technische Realisierung festgelegter Begriff.

5.1.2 Abstraktionsebenen des Modells - Verfeinerung / Verschattung

Ein wichtiges Konzept des Modells und anderer Modellierungstechniken ist die Verfeinerung oder die Verschattung von Elementen. Eine Verfeinerung ist das Hinzufügen von detaillierteren Modellelementen zu einem vorhandenen Modell. Der Übergang von einer Black-Box Sicht einer Komponente hin zu einer Glass-Box Sicht entspricht genau diesem Vorgang. Das Modell wird so detaillierter und weitere Bestandteile sichtbar. Durch Verfeinerung ist eine Abstraktion möglich, die nur die jeweils für die aktuelle Anforderung notwendige Detaillierungsstufe des Modells darstellt. Darüber hinaus kann durch dieses Vorgehen ein Modell iterativ entworfen und erweitert werden. Die Strukturierung eines Modells ist damit einfacher möglich.

---

[40] Entsprechende Änderungen sind auch über den Lauf eine Jahres hinweg zur Optimierung der Erstellung der Bilanz, für Lohnabrechnung oder andere zyklische Dienste denkbar, um mit einem geringeren Einsatz an IT-Ressourcen durch die gemeinsame Nutzung einer Infrastruktur Kosten zu verringern.

[41] Wir verwenden die Begriffe IT-Service und IT-Dienst gleichbedeutend

Durch die Verfeinerung werden die internen Strukturen einer Komponente sichtbar. Wird ein IT-System verfeinert, indem es bei einer Glass-Box-Dekomposition um zwei virtuelle Instanzen unterteilt wird, so wird bei dieser Aufspaltung auch die Verteilung der Softwarekomponenten des Gesamtsystems auf die virtuellen Instanzen sichtbar, wenn die Zuordnung dieser Softwarekomponenten zu den virtuellen Maschinen modelliert wird. Dasselbe gilt für die Endpunkte und die daran gebundenen Schnittstellen. Wird in einem Server ein Speichersubsystem modelliert, so kommen zum Modell der IT-Infrastruktur zusätzliche fail-Funktionen[42] hinzu, die dem Modell zusätzliche Informationen über seine Teilkomponenten hinzufügen.

Das Gegenteil der Verfeinerung ist die Verschattung. Sie beschreibt den Übergang von einer Glass-Box Sicht hin zu einer Black-Box Sicht einer Komponente. Dies kann sinnvoll sein, wenn Modelle zu umfangreich für eine vereinfachte Darstellung werden oder für eine spezielle Sicht Detailinformationen nicht notwendig sind. Darüber hinaus kann dadurch die Modellierung von komplexen Systemen vereinfacht werden, indem zuerst ihre Bestandteile modelliert werden und diese dann zu einem Modell des Gesamtsystems integriert werden. Wird ein modellierter Server mit einem Speichersubsystem dahingehend verschattet, dass das Speichersubsystem unsichtbar wird, so wird die entsprechende fail-Funktion des Server entsprechend der fail-Funktion des Speichersubsystems erweitert. Bei der Verschattung können Informationen verloren gehen. Dieser Vorgang kann möglicherweise nicht automatisch revidiert werden, sondern es müssen möglicherweise Informationen dem Modell wieder manuell hinzugefügt werden.

Die verschiedenen Funktionen und Auswirkungen von Verfeinerung und Verschattung werden bei den verschiedenen Modellelementen beschrieben. Die Abbildung 34 verdeutlicht die Verfeinerung und die Verschattung.

5.1.3 Instanzen

Verschiedene Elemente eines IT-Dienstes, wie Softwarekomponenten, können mehrmals innerhalb der IT-Infrastruktur vorhanden sein und auf mehreren Servern ausgeführt werden. Ein Beispiel sind Softwarekomponenten wie Betriebsysteme, die auf verschiedenen Systemen ausgeführt werden. Aber auch die IT-Dienste an sich sind möglicherweise mehrfach vorhanden. So könnte ein IT-Dienstleister einen Dienst für verschiedene Kunden erbringen. In diesem Fall ist es wichtig, diese mehrmals vorhandenen Objekte beschreiben und eindeutig identifizieren zu können. Wir führen dazu Instanzen ein. So lassen sich IT-Dienste modellieren, deren Instanzen zum Beispiel die verschiedenen Kunden beschreiben, für die der IT-Dienst betrieben wird.

Instanzen existieren dabei für alle Objekte und helfen dabei, verschiedene Objekte desselben Typs zu beschreiben. Wir definieren eine abstrakte Default-Instanz, die als Platzhalter fungiert, solange keine spezielle Instanz angegeben ist. Darüber hinaus zeichnet sich die Default-Instanz dadurch aus, dass durch sie alle Objekte desselben Typs beschrieben sind. Werden Attribute der Default-Instanz beschrieben, gelten diese Attribute auch für alle benannten Instanzen.

> *Instanzen beschreiben ein existierendes Objekt in einem Teilmodell von einem Typ. Die Instanzen sind dabei mit einem Bezeichner spezifiziert. Für jedes Objekt existiert eine Default-Instanz. Die konkreten Instanzen können die Attribute der Default-Instanz verschatten.*

Definition 20 – Instanzen

Wichtig für die Konsistenz des Modells ist, dass Instanzen eindeutig referenzierbar und unterscheidbar sind. Durch unsere Instanzen wird dies sichergestellt. Es können in einem Modell nicht zwei gleiche Softwarekomponenten mit demselben Instanznamen existieren. Mit dem Instanznamen wird eine der beiden identischen Softwarekomponenten wieder eindeutig referenzierbar. Dies gilt auch für alle anderen Elemente der anderen Teilmodelle.

[42] Wir führen die Fail Funktionen in Abschnitt 5.5.5 ein

Instanz = Objekt[Instanzbezeichner]

create(Objekt [, Instanzbezeichner]) = Objekt[Instanzbezeichner]

create(Objekt1)=Objekt1[default]

create(Objekt2,“A“)=Objekt2[A]

Formel 1 – Funktionen auf Instanzen

Die Instanz wird in eckigen Klammern direkt hinter dem jeweiligen Objekt angegeben. Bei der Default-Instanz kann diese Schreibweise vereinfacht und die Klammern ausgelassen werden.

Wir definieren die Funktion Instanzzahl, die uns die Zahl der Instanzen eines Objektes angibt.

instanzzahl(create(Objekt1))=1

instanzzahl(create(Objekt2,"A"))=1

instanzzahl(create(Objekt1)) ∧ create(Objekt1,"A"))=1

instanzzahl(create(Objekt1,"A")) ∧ create(Objekt1,"B"))=2

Formel 2 – Instanzzahl von Objekten

Die Funktion ergibt als Instanzzahl 1, falls eine Default-Instanz oder eine benannte Instanz des Objektes existiert. Wird eine Default-Instanz und eine konkrete Instanz eines Objektes angelegt, so dient die Default-Instanz nur noch zur Beschreibung aller Instanzen dieses Objekttyps, die Instanzzahl ist also weiterhin 1, weil die Default-Instanz im Modell nicht mehr existiert sondern nur noch als Platzhalter verwendet wird.

Wir werden bei den einzelnen Modellelementen auf die Eigenschaften und Ausprägungen der Instanzen zurückkommen und diese für den jeweiligen Typ beschreiben.

5.1.4 Schnittstellen

Schnittstellen beschreiben in unseren Teilmodellen die Übergabepunkte von Informationen und stellen die Basis von Kanälen dar. Schnittstellen existieren in unserem Softwaremodell und im Infrastrukturmodell. Sie definieren innerhalb der Softwaremodelle Übergangspunkte und Nachrichtenformate zwischen verschiedenen Softwarekomponenten und damit die Syntax für den Austausch von Informationen. Eine Entsprechung in der Softwareentwicklung sind IDLs wie in CORBA oder Microsoft COM, bzw. neuere Techniken wie WSDL-Beschreibungen von Webservices. Sie sind an die jeweilige Softwarekomponente, die die Schnittstelle erbringt oder konsumiert, gebunden. Innerhalb unseres Softwaremodells ist es jedoch nicht notwendig, binäre Nachrichtenformate oder XML Daten zu beschreiben. Eine abstrakte Modellierung ist ausreichend und erfolgt durch entsprechend gewählte Bezeichner. Damit ist die Modellierung der Beziehung einfach möglich, stützt sich aber trotzdem auf der technischen Erbringung ab.

Im Infrastrukturmodell beschreiben Schnittstellen die Bindung an ein IT-System bzw. die IT-Infrastruktur und dessen Endpunkte sowie die jeweilige Instanz des Softwaresystems. Damit sind Instanzen von Softwarekomponenten über ihre Schnittstelle mit der jeweiligen Instanzbeschreibung unterscheidbar und innerhalb der Teilmodelle eindeutig. Der Infrastrukturanteil einer Schnittstelle wird bei der Installation der Softwarekomponenten auf dem jeweiligen IT-System in der Schnittstellenbindung des Endpunktes definiert, in diesem Schritt werden in der Schnittstelle modellierte Instanzbeschreibungen vervollständigt. Zur Vereinfachung von Schnittstellen lassen wir es zu, dass je nach Anwendung Instanzbeschreibungen bei der Bindung an IT-Systeme hinzugefügt werden können, wenn sie für den jeweiligen Schnittstellenbezeichner sinnvoll sind. Dadurch müssen nicht an jeden Schnittstellenbezeichner bereits bei der Modellierung Instanzbezeichner vorgesehen werden.

Im Gegensatz zu Schnittstellen, wie sie in der Softwareentwicklung verwendet werden, sind die Schnittstellen unseres Modells abstrakter gehalten. Die beiden beschriebenen Anteile einer Schnittstelle finden sich in der Schnittstellenhierarchie. Eine feste Trennung der Anteile ist hier nicht sinnvoll und wird nicht vorgenommen, es hängt von der Wahl des Hierarchiestacks ab,

welche Anteile durch Hardware und welche durch Software realisiert werden. Diese Unterscheidung hängt teilweise auch von der eingesetzten Hardware ab. Aktuelle Serversysteme und Netzwerkkarten ermöglichen mit TCP/IP Offload-Engines, einen Teil des eigentlich in Software implementierten TCP/IP Stacks direkt auf der Netzwerkkarte auszuführen und nicht innerhalb des Betriebsystems.

Ein Beispiel für eine Schnittstelle mit einer Hierarchie, wie schon im vorigen Kapitel eingeführt, ist gegeben durch:

Ethernet : TCP/IP : Webserver

Diese Schnittstelle beschreibt vereinfacht die Schnittstelle eines Webservers. In diesem Beispiel wird die erste Hierarchieebene, Ethernet, realisiert durch die Netzwerkkarte im Server. Diese Ebene ist also dem Infrastrukturmodell zuzuordnen. TCP/IP wird in heute gängigen Systemen durch das Betriebsystem erbracht, ist also dem Softwaremodell zuzuordnen (Ausnahme TCP/IP Offloading). Gleiches gilt für den Webserver, der die letzte Hierarchiestufe realisiert und meist als eigene Softwarekomponente in einem Betriebsystem ausgeführt wird. Die Unterscheidung der Ebenen ist im Weiteren nicht wirklich notwendig, wichtig ist, dass eine vollständig definierte Schnittstelle mit einem Softwareanteil automatisch das Softwaremodell und das Infrastrukturmodell umfasst.

Wir definieren unsere Schnittstelle S in BNF nach [Broy98]

*Schnittstellen definieren Datenübergabepunkte innerhalb unseres Modells. Sie teilen sich in Hierarchien auf, wobei jede Hierarchieebene durch einen eindeutigen Bezeichner gegeben ist. Schnittstellen werden gebunden an Endpunkte. Damit werden die Softwarekomponenten, die höhere Hierarchieebenen der Schnittstelle erbringen, an IT-Infrastrukturkomponenten gebunden, die die physikalischen Schichten der Schnittstellenhierarchie realisieren. Während der Modellierung ist der Platzhalter * erlaubt, der eine Unterspezifikation der Schnittstelle beschreibt und bei der Bindung an ein IT-System durch die konkreten Werte ersetzt wird.*

Definition 21 – Schnittstelle

Die Syntax der Schnittstellen ist gegeben durch:

Instanzbezeichner = Bezeichner { | Bezeichner }

Schnittstellenbezeichner = Bezeichner | Bezeichner[Instanzbezeichner] { „|“ Bezeichner | „|“ Bezeichner[Instanzbezeichner] }

*Schnittstellenhierarchie = { *: } Schnittstellenbezeichner { : Schnittstellenbezeichner }* { :* }*

Schnittstelle S = (Schnittstellenhierarchie)

Formel 3 – Syntax von Schnittstellen

Schnittstellen bauen die Schnittstellenhierarchien auf, wie in Abbildung 45 dargestellt. Eine einfache Schnittstellendefinition ist die in Formel 4 definierte http Webschnittstelle S2. Dabei ist der Schnittstellenbezeichner in einem Stack von links nach rechts zu lesen. Wie bereits beschrieben können beliebige Bezeichner für komplexe Protokolle und Nachrichtentypen verwendet werden, die aber für alle Teilmodelle eindeutig sein müssen.

Der Bezeichner * beschreibt beliebige Protokolle, die noch nicht spezifiziert sind. Wir erlauben eine Unterspezifikation des Protokollstacks[43]. Um Aussagen über den Detaillierungsgrad einer Schnittstelle machen zu können, führen wir die Funktion Hierarchiezahl ein, die die Anzahl der Hierarchien einer Schnittstelle angibt.

[43] Dies erleichtert zum einen die Modellierung, wird aber auch für Schnittstellen auf der Softwareebene benötigt, die noch nicht an eine Infrastrukturkomponente gebunden sind. Bei der Bindung erfolgt dann die Auflösung und Vervollständigung der Schnittstellenhierarchie.

Hierarchiezahl(S) = hz=0;

for each Schnittstellenbezeichner(S) {

if Schnittstellenbezeichner="" then hz+0 else hz+1;*

}

return hz;

Beispiel einer Softwarekomponente, die ein http Schnittstelle anbietet:

S2=(:http)*

Hierarchiezahl(S2) = 1

Formel 4 – Hierarchiezahl von Schnittstellen

Wie aus der Formel erkennbar, gehen nur spezifizierte Hierarchien in die Bewertung mit ein. Der Platzhalter * zählt nicht als Hierarchieebene[44]. Dies gilt später auch für die Funktionen auf Schnittstellen.

Die möglichen Hierarchien einer Schnittstelle werden durch das Modell nicht festgelegt. Für einen konkreten Fall ist die Definition einer Hierarchie aber sinnvoll, um die Modellierung von nicht sinnvollen Schnittstellen zu vermeiden, wie in Formel 5. Eine Kapselung von Token-Ring Paketen in einem Ethernetframe ist zwar möglich und tritt in Migrations- oder Legacyszenarien möglicherweise auf, der Transport von Ethernetpakten über http ist aber sinnlos.

S4 = (http:ethernet:token-ring:ip)

Formel 5 – Nicht sinnvolle Schnittstelle

Häufig verwendete Protokolle wie die IP Familie lassen sich so direkt nach dem bekannten ISO-OSI Modell in eine Hierarchie einordnen, die sich direkt anwenden lässt. Sie kann dann bei der Modellierung entsprechend als Hilfe verwendet werden. Abbildung 45 verdeutlicht dies an mehreren Beispielen.

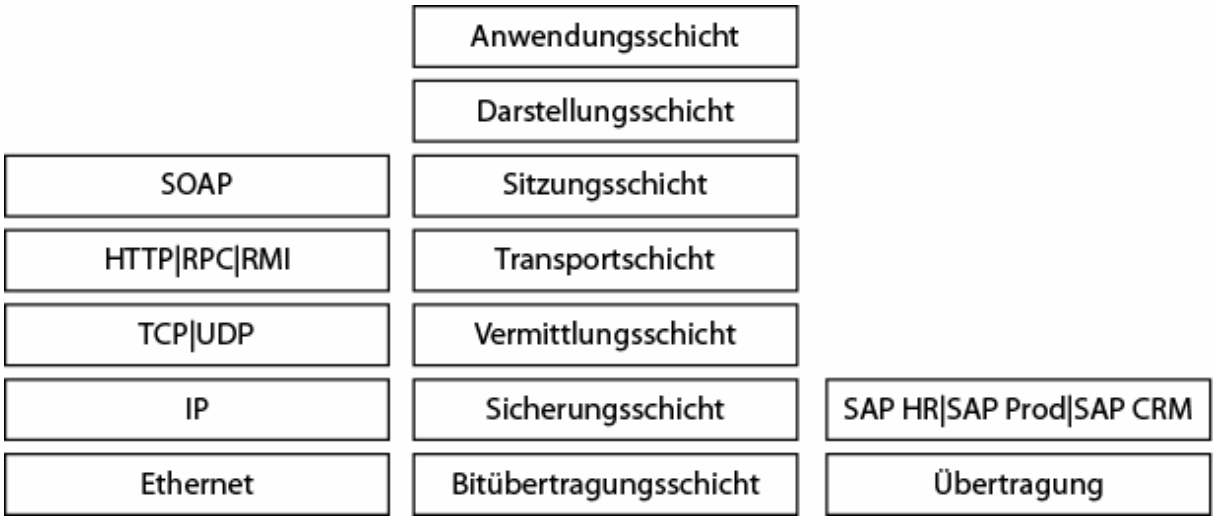

Abbildung 45 – Beispiele für Schnittstellenhierarchien

[44] Dies entspricht der Default-Instanz, auch diese führen wir nur als Platzhalter für die konkrete Instanz ein

In der Mitte ist das bekannte ISO-OSI Referenzmodell dargestellt. Die entsprechende Schnittstellendefinition in unserer Syntax gibt Definition 22 an.

Eine Schnittstellenhierarchie ist gegeben durch:

(Bitübertragung : Sicherung : Vermittlung : Transport : Sitzung : Darstellung : Anwendung)

Definition 22 – ISO-OSI Hierarchie (nach [Tan03], S56)

Links findet sich eine beispielhafte Hierarchie für heute gängige Webservicebasierte Architekturen wie Microsoft ASP.NET. Für manche Modelle kann aber auch eine vereinfachte Hierarchie, wie rechts dargestellt, ausreichend sein. Soll nur die Verteilung einer bestimmten Klasse von Anwendungen dargestellt werden und andere Informationen abstrahiert werden, kann die Verwendung einer entsprechend vereinfachten Hierarchie das Modell entlasten und vereinfachen. Hier ergibt sich auch eine Möglichkeit zum Tailoring des Modells[45].

Auf den Modellhierarchien lassen sich auch Abbildungen angeben, die zwischen verschiedenen Hierarchien übersetzen können. Damit ist die Verwendung von unterschiedlichen Hierarchien innerhalb eines Modells möglich. So können verschiedene Topologien ineinander übersetzt werden. Ein Bespiel dafür ist der Übergang aus einer SAN basierten Fibre-Channel Umgebung hin zu einer Ethernet Umgebung. Die physikalische Komponente, die den Übergang zwischen den Modellbereichen realisiert, wäre dann der Ort der Übersetzung. Wichtig ist hierbei, eine Aussage über die Vergleichbarkeit von Schnittstellenhierarchien treffen zu können. Dies ist vor allem wichtig, wenn Endpunkte mit ihrer Schnittstellenhierarchie über einen Kanal miteinander verbunden werden sollen. Wir definieren dazu die Funktion compat.

Schnittstellehierarchie = Schnittstellenbezeichner x Schnittstellenhierarchie

first(Schnittstellenhierarchie) = <Schnittstellenbezeichner> ° Schnittstellenhierarchie'

mit Hierarchiezahl(Schnittstellenhierarchie') = Hierarchiezahl (Schnittstellenhierarchie) - 1

last(Schnittstellenhierarchie) = Schnittstellenhierarchie' ° <Schnittstellenbezeichner>

mit Hierarchiezahl(Schnittstellenhierarchie') = Hierarchiezahl (Schnittstellenhierarchie) -1

Schnittstellenhierarchie = first(Schnittstellenhierarchie) $\cap$ *rest(Schnittstellenhierarchie)*

Beispiel: first (S3[46]*) = „IPsec|PPTP" ° „*:SOAP:Webservice1"*[47]*

Schnittstellenhierarchie x Schnittstellenhierarchie = bool

compat (SH1, SH2)=

(true, falls first(SH1) = first(SH2) $\vee$

false, falls first(SH1) $\neq$ *first(SH2))* $\wedge$

compat(rest(SH1),rest(SH2))

Formel 6 – Funktionen auf Schnittstellenhierarchien

[45] Wir gehen auf das Tailoring im Kapitel 6.2 ein

[46] Siehe Definition 23

[47] Der Bezeichner * zählt nicht als Hierarchieebene. Die beiden entstehenden Bezeichner „*:*“ werden zu „*“ zusammengefasst.

Die Funktion first liefert den Bezeichner der untersten Schnittstellenhierarchieebene, das Gegenstück dazu ist die Funktion last, die das oberste Element liefert[48]. Die Funktion compat vergleicht zwei Schnittstellenhierarchien rekursiv, ob die Bezeichner auf der jeweiligen Ebene identisch sind und die Schnittstellen damit demselben Schnittstellenstack entsprechen. Die verschiedenen Instanzen werden dabei nicht verglichen, da unterschiedliche Instanzen trotzdem vom selben Typ sind.

Unsere Definition von Schnittstellen erlaubt auch die Festlegung von Optionen. Damit wird es möglich, auf einer Ebene mehr als ein Protokoll zu spezifizieren.

S3 = (:IPsec|PPTP:*:SOAP:Webservice1)*

Definition 23 – Schnittstellendefinition mit Optionen

Die Schnittstellendefinition Definition 23 legt fest, dass die Verbindung zu dieser Schnittstelle über eines der beiden gesicherten Protokolle IPsec oder PPTP geführt werden muss. Ob die Daten auf der Hardwareschnittstelle über Ethernet, Token Ring oder andere Protokolle übertragen werden, wird durch diese Schnittstelle nicht definiert. Darüber hinaus ist auch nicht spezifiziert, ob der Zugriff auf den Webservice per http oder aber ein anderes Protokoll erfolgt.

Wichtig ist diese Art der Modellierung vor allem für Softwarekomponenten, bevor diese auf Infrastrukturkomponenten zugewiesen werden. Damit lassen sich Anforderungen einer Softwarekomponente an die Netzwerkinfrastruktur modellieren. Die Vorgaben dazu können auch durch einen Geschäftsprozess gegeben sein, wenn dieser entsprechende Sicherheitsanforderungen an die IT-Infrastruktur stellt. Bei der Zuweisung einer Softwarekomponente an ein IT-System wird dann eine mögliche Option ausgewählt oder die Schnittstelle wird in zwei Schnittstellen aufgespaltet, die dann dem Endpunkt des jeweiligen IT-Systems zugeordnet werden. Bei einer zugewiesenen Schnittstelle auf eine Infrastrukturkomponente ist die Zahl der Optionen damit immer 0. Die Funktion Optionszahl gibt die Zahl der unterschiedlichen Optionen an, die in einer Schnittstelle noch vorhanden sind:

Optionszahl(Schnittstelle S)=Instanzzahl(S) - Hierarchiezahl(S)

Formel 7 – Optionszahl: Anzahl der Optionen einer Schnittstelle

Die Optionszahl ergibt sich aus der Anzahl der Instanzen einer Schnittstelle. Pro Hierarchie ist mindestens eine Instanz vorhanden. Dabei handelt es sich entweder um die Default-Instanz oder eine konkrete Instanz. Damit ergibt sich die Zahl der freien Optionen als die Zahl der Instanzen abzüglich der Zahl der Hierarchien.

Die Instanziierung von Diensten wird wie bei IT-Systemen durch eine Verfeinerung der Schnittstelle beschrieben. Dabei werden die Instanzbezeichner auf die entsprechenden Werte gebunden. Ein einfacher Webservice-basierter Rechendienst, dessen ausgeführte Operation bei der Installation festgelegt wird, könnte die Schnittstelle wie in Formel 8 implementieren. Die verschiedenen Möglichkeiten für die Instanzbeschreibung werden dabei in eckigen Klammern angegeben. Falls die Information nicht festgelegt werden soll, können die Klammern auch weggelassen werden[49]. Installiert der Administrator diesen Dienst, wird die Operation der Instanz festgelegt. Damit wird die Schnittstelle verfeinert.

[48] Der Vergleich bezieht sich dabei auf die immer vorhandene Default-Instanz

[49] Bei der Modellierung ist zu beachten, dass sich hierdurch eventuell Mehrdeutigkeiten ergeben können, wenn die möglichen Optionen nicht spezifiziert sind

S4 = (:http:soap:webcalc[plus|minus|mal|geteilt])*

Optionszahl(S4)=4

Nach Installation als Instanz Plus auf einem konkreten Server[50]:

S4´ = (Ethernet[00:00:00:00:00:00]:ip[192.168.6.100]:tcp[80]:http[get]: soap[1.0]:webcalc[plus])

Optionszahl(S4´) = 0

Formel 8 – Schnittstelle mit Instanzinformationen

Die Formel verdeutlicht dabei, dass auch Protokolle entsprechend instantiiert werden können. So wird beim Protokoll tcp der Port 80 verwendet, auf Ethernetebene ist die Instanz an die MAC Adresse des IT-Systems gekoppelt. Nach der Bindung sind alle Optionen entsprechend definiert, die Optionszahl ist damit 0. Wir definieren Funktionen zum Umgang mit Instanzinformationen.

getInstanzinformationen(Schnittstellenbezeichner SB1) = <Instanzbezeichner>

addInstanzinformation(Schnittstellenbezeichner SB1, Instanzbezeichner IB1) →

SB1=SB1[getInstanzinformation(SB1) ∩ IB1]

removeInstanzinformation(Schnittstellenbezeichner SB1, Instanzbezeichner IB1) →

∀ Instanzbezeichner in getInstanzinformationen(SB1) gilt:

Instanzbezeichner <> IB1

Formel 9 – Funktionen für Instanzinformationen auf Schnittstellen

Um Schnittstellen und ihren Detailgrad miteinander vergleichen zu können, benötigen wir die Funktion InstanzzahlSchnittstelle, die uns alle Instanzen einer Schnittstellenhierarchie gibt. Damit werden nicht nur die Instanzen einer einzelnen Schnittstellenhierarchien, sondern aller Ebenen addiert.

InstanzzahlSH(Schnittstelle S)=

Instanzzahl(first S) + InstanzzahlSH (first(S))

Formel 10 – InstanzzahlSH: Anzahl der Instanzen einer Schnittstellenhierarchie

Durch die Funktion Instanzzahl wird die Zahl der Hierarchien ohne Instanzbezeichner mit eins gezählt, da eine Defaultinstanz auf jeder Hierarchieebene vorhanden ist.

Die Funktion zur Verfeinerung spezifiziert zusätzliche Ebenen der Schnittstelle. Im Beispiel wird die bereits oben eingeführte Schnittstelle S2 dabei von beliebigen http Anfragen auf eine konkrete Webanwendung (Webanwendung1) verfeinert.

[50] Für die Funktionen, die bei der Installation verwendet werden, siehe Kapitel 5.6

Verfeinerung(Schnittstelle S,Schnittstellenhierarchie) = Schnittstelle S´

S2=(:http)*

Verfeinerung(S2,":http:Webanwendung1") → S2´= (*:http:Webanwendung1)*

Dabei gilt:

Hierarchiezahl(S2) <= Hierarchiezahl(S2´), 1 < 2

InstanzzahlSH(S2) <= InstanzzahlSH(S2´)

Optionszahl(S2) <= Optionszahl(S2´)

und Hierarchiezahl(S2) + InstanzzahlSH(S2) + Optionszahl(S2) <

Hierarchiezahl(S2´) + InstanzzahlSH(S2´) + Optionszahl(S2´)

und Schnittstellenbezeichner S Teilmenge S¹

Formel 11 – Verfeinerung von Schnittstellen

Die inverse Funktion zur Verfeinerung einer Schnittstellenhierarchie ist das Entfernen von Hierarchieinformationen.

Entfernen(Schnittstelle S,Schnittstellenhierarchie) =Schnittstelle S´

S2'=(:http:Webanwendung1)*

Verfeinerung(S2,":Webanwendung1") → S2= (*:http)*

Dabei gilt:

Hierarchiezahl(S2) >= Hierarchiezahl(S2´), 1 < 2

InstanzzahlSH(S2) >= InstanzzahlSH(S2´)

Optionszahl(S2) >= Optionszahl(S2´)

und Hierarchiezahl(S2) + InstanzzahlSH(S2) + Optionszahl(S2) >

Hierarchiezahl(S2´) + InstanzzahlSH(S2´) + Optionszahl(S2´)

und Schnittstellenbezeichner S¹ Teilmenge S

Formel 12 – Entfernen von Hierarchieinformationen

Schnittstellen bilden zusammen mit Endpunkten und Kanälen die Kommunikationsbeziehungen in unseren Teilmodellen ab. Auf diese weiteren Elemente gehen wir in den nächsten Abschnitten ein.

5.1.5 Endpunkte

Endpunkte definieren Übergabepunkte von Daten in unserem IT-Dienstmodell. Sie beschreiben die Verbindung von IT-Systemen mit der Kommunikationsinfrastruktur. Endpunkte sind deshalb gebunden an ein oder mehrere IT-Systeme oder an andere Endpunkte. Sie entsprechen technisch Netzwerkkarten oder Ports von IT-Infrastrukturkomponenten wie einem Ethernet Switch. Endpunkte besitzen mindestens eine Schnittstelle, deren Funktionalität sie anbieten und sind über Kanäle miteinander verbunden. Die Schnittstellenhierarchie eines Endpunktes muss daher auf der ersten Ebene immer eindeutig sein, der Bezeichner * ist nicht erlaubt, um eine eindeutige Identifikation des Endpunktes auf dem Kanal sicherzustellen. Dies entspricht dem Konzept der weltweit gültigen MAC/WWNID Adressen einer Netzwerkkarte.

Die Möglichkeit, einen anderen Endpunkt als eine Endpunktbindung für einen Endpunkt angeben zu können ermöglicht die Modellierung von Load-Balanced-Konfigurationen. Diese Lösungen zeichnen sich dadurch aus, dass mehrere Server gemeinsam einen Dienst anbieten. Dadurch lässt sich die Last auf mehrere Server verteilen. Diese Technik wird häufig bei Webser-

vern oder aber auch bei Media-Streaming Servern eingesetzt, wichtig ist, dass die Server keinen internen Zustand beinhalten, der eine Verteilung der Datenbasis notwendig machen würde[51]. Ein gemeinsamer Zugriff auf die Datenbasis würde einen Mechanismus benötigen, der sicherstellt, dass alle Änderungen an den Daten konsistent durchgeführt werden. Fällt ein System aus, übernehmen die anderen Server die Clients des ausgefallen Server. Für einen Benutzer ist dieses Verhalten völlig transparent, so lange insgesamt genügend Ressourcen zur Verfügung stehen. In unserer Modellierung werden in diesem Fall alle beteiligten physikalischen Endpunkte der jeweiligen Systeme als Endpunktbindungen für den virtuellen Load-Balanced Endpunkt angegeben. Dieser realisiert dann die verteilte Schnittstelle, die den entsprechenden Dienst zur Verfügung stellt. An diesen Endpunkt muss anschließen das Verhalten der Lastverteilungslösung in der fail() Funktion modelliert werden.

Ein Endpunkt beschreibt die Verbindung zwischen mindestens einem IT-System und der Kommunikationsinfrastruktur. Dazu wird der Endpunkt über eine Endpunktbindung mit dem IT-System verbunden. Alle Schnittstellen, die ein IT-System über diesen Endpunkt anbietet, werden als Schnittstellen dieses Endpunktes modelliert. Zur Modellierung von HA-Lösungen ist auch eine Endpunktbindung zu einem anderen Endpunkt möglich. Den entstehenden Endpunkt bezeichnen wir einen virtuellen Endpunkt.

Definition 24 – Endpunkt

Der Endpunkt ist die Verbindung zwischen dem IT-System und den Netzwerkkomponenten und beinhaltet als Container die Schnittstellen, die an diesen Endpunkt gebunden sind. Die Syntax des Endpunktes und der Endpunktbindung lautet:

Endpunktbindung EPB = (IT-System[52] | Endpunkt [{ ,IT-System | ,Endpunkt }])*

Endpunkt EP = (Endpunktbindung : { Schnittstelle })*

Formel 13 – Syntax von Endpunkten

Die Endpunktbindung und der Endpunkt existiert in unserem Modell nicht als eigenständiges Objekt, sondern sie wird in eine Instanz eines IT-Systems umgewandelt[53]. Die Funktionen definieren einen Endpunkt. Die übergebene Schnittstellenhierarchie muss innerhalb des Modells eindeutig definiert sein. Wir lassen auf der ersten Ebene der Schnittstellenhierarchie, wie bereits beschrieben, keine Optionen zu.

Bezeichner → Endpunkt

createEP(Schnittstellenhierachie S1) = EP1

createEP(„[00:00:00:00:00:00]") = EP1

Sei SH die Menge aller Schnittstellen

Dabei gilt: $\forall SH_a \in SH: SH_a \neq SH_1$

Optionszahl(SH1) = 0

Formel 14 – Funktion zum Anlegen eines Endpunktes

Wir definieren Funktionen auf Endpunkten, die diese an IT-Systeme binden und ihnen Schnittstellen zuordnen.

[51] Sobald eine zentrale Datenhaltung wichtig wird, muss hier meist auf externe Lösungen zugegriffen werden

[52] Wir verzichten im Rahmen unseres Modells auf die Modellierung von Instanzen von IT-Systemen, siehe Kapitel 5.5

[53] Siehe dazu Instanzen auf IT-Systemen in Kapitel 5.5

BindeEndpunkt (Endpunkt EP1, Endpunktbindung EPB) → Endpunkt EP1´
mit Zahl der Endpunktbindungen (EP1) < Zahl der Endpunktbindungen (EP1')

EntferneBindung (Endpunkt EP1', Endpunktbindung EPB) → Endpunkt EP

mit Zahl der Endpunktbindungen (EP1) > Zahl der Endpunktbindungen (EP1')

BindeSchnittstelle (Endpunkt EP1, Schnittstelle S1) → Endpunkt EP1"

mit Zahl der Schnittstellen (EP1) < Zahl der Schnittstellen (EP1')

und Optionszahl(S1)=0

EntferneSchnittstelle (Endpunkt EP1", Schnittstelle S1) → Endpunkt EP1

mit Zahl der Schnittstellen (EP1) > Zahl der Schnittstellen (EP1')

Formel 15 – Funktionen auf Endpunkten

Bei der Bindung der Schnittstelle werden alle Optionen auf den verschiedenen Schnittstellenhierarchien an eine konkrete Option gebunden. Werden mehrere Optionen benötigt, wird die Schnittstellenhierarchie doppelt mit jeweils einer ausgewählten Option gebunden.

Für die Modellierung eines Clustersystems wird wie bei einer Load-Balanced Konfiguration ein Virtueller Endpunkt eingeführt, der an die jeweiligen physikalischen Endpunkte der verschiedenen Clusterknoten gebunden wird. Je nach Funktionsweise des Clusters erfolgt die Bindung an einen oder an alle Knoten. Clustersysteme entsprechen in ihrer grundlegenden Funktionsweise Load-Balanced Konfigurationen. Clusterlösungen werden immer dann eingesetzt, wenn nicht nur die Rechenleistung oder die Netzleistung eines Servers wichtig ist, sondern vor allem bei Diensten, die auf einem Datenbestand arbeiten. Ein sehr klassisches Beispiel für Cluster sind Datenbank- oder Emailsysteme. Im Gegensatz zu Load-Balanced Lösungen ist hier ein Zustand des Servers, also der Datenbestand, der Dienst, der angeboten wird. Um eine Replikationsmechanismus und den gleichzeitigen Zugriff von zwei Servern mit all den verbundenen Schwierigkeiten zu vermeiden, wird hier der Dienst von nur einem Server erbracht. Fällt dieser aus, wird der Dienst auf einen anderen Server verlagert. Man spricht hier von Aktive:Passive Clustering, da nur ein Server den Dienst ausführt. Der Dienst wird immer nur von einem Server bereitgestellt. Wichtig ist in diesem Fall ein gemeinsames Speichersystem für alle Clusterknoten. Die Fail-Funktion des virtuellen Endpunktes muss auch in diesem Fall die Logik der Clusterkonfiguration, wie das Umschaltverhalten, im Modell abbilden. Wir geben dazu im Kapitel 7.3 ein Beispiel an.

Die Verwendung einer Instanz eines IT-Systems erlaubt die Modellierung von IT-Systemen mit mehreren Netzwerkkarten. Dies entspricht dem Verständnis, dass ein IT-System in unserem Modell durch seine Schnittstelle nach außen, also die Ethernetkarte spezifiziert ist[54]. Jedes IT-System hat per Definition einen internen Endpunkt, der an das IT-System selbst gebunden und mit keinem Kanal verbunden ist. Dies entspricht dem von IP bekannten Konzept der localhost oder 127.0.0.1 Schnittstelle. Dies unterscheidet sich aber von unserem Verständnis der Default-Instanz. Diese ist NICHT mit dem internen Endpunkt identisch, da diese ja bei der Instanziierung nicht mehr vorhanden wäre.

Dieser Endpunkt beschreibt alle Schnittstellen, die für das IT-System verwendet werden.

E = (Server1[localhost] : IP[127.0.0.1]:TCP:SMB ,
Server1[localhost] : IP[127.0.0.1]:UDP:SMB)

Formel 16 – Server mit Windows Dateifreigabe[55] an Localhost

[54] Es wäre auch möglich, die IT-Systeme als eine zusätzliche Ebene in der Schnittstellenhierarchie einzuführen. Wir verzichten zur Vereinfachung darauf. Dies entspricht auch eher dem Verständnis und der Sicht eines Administrators

[55] Kommunikation über TCP und UDP mit dem SMB Protokoll

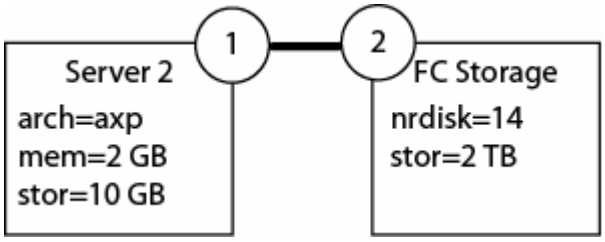

Abbildung 46 – Grafische Darstellung von Endpunkten (Server-FC Storage)

Die Abbildung 46 zeigt einen Server, der an ein Fibre channel Speichersubsystem angeschlossen ist. Die Endpunkte sind durch einen Kreis symbolisiert. In diesem Fall definieren die Endpunkte 1 und 2 zusammen mit dem Glasfaserkabel, dem durch die Linie gekennzeichneten Kanal, das Fibre channel Netzwerk[56]. An den Endpunkt 1 des Servers 2 wäre in diesem Fall die Schnittstelle aus Formel 17 gebunden.

S7 = (Server2[10:00:00:00:00:00:00:00]:Fibrechannel)

I = (Server2)

E = (I , S7)

Formel 17 – Fibrechannel-Loop[57] Schnittsstellendefinition

Die einzelnen, physikalischen Netzwerkkarten finden sich in den Instanzen des Servers, der in der Schnittstelle die entsprechende WWNID als Instanzinformation beinhaltet[58].

5.1.6 Attribute

Alle Elemente unserer Teilmodelle können durch Attribute beschrieben und um Informationen angereichert werden. Wir unterscheiden zwei Arten von Attributen. Statische Attribute oder Schlüssel beschreiben Schlüssel-Wertepaare, die eine Eigenschaft eines Systems oder einer Komponente genauer spezifizieren. Diese Attribute sind an sich statisch, können sich aber über die Zeit verändern.

Dynamische Attribute werden repräsentiert durch Funktionen, die Ihren Wert und den entsprechenden Verlauf beschreiben. Sie eignen sich vor allem, um Attribute oder Anforderungen zu beschreiben, die sich über die Zeit kontinuierlich ändern. Ein Beispiel ist die Beschreibung des Lastprofils einer Anwendung. Diese lässt sich als Funktion aufstellen und der jeweiligen Komponente zuordnen.

Attribute werden gekennzeichnet durch einen Schlüssel, über den der Zugriff erfolgt, und einen Wert. Dieser Wert kann auch durch eine Funktion gegeben sein.

Definition 25 – Attribute

Die Syntax der Attribute ist angelehnt an die Verwendung von Attributen in objektorientierten Programmiersprachen:

Server1.memory=2 GB

Formel 18 – Syntax von Attributen

[56] In diesem Fall einen Fibre channel Loop ohne aktive Netzwerkkomponenten

[57] Die Formel vereinfacht in diesem Fall den Fibre channel Stack und kennzeichnet die entsprechende Schnittstelle als Fibre channel Netz, um sie von einer Ethernetschnittstelle unterscheidbar zu machen. Eine vollständige Modellierung des Fibre Chanel Stack ist ebenfalls möglich.

[58] Siehe dazu Instanzen auf IT-Systemen in Kapitel 5.5

Statische Attribute/Schlüssel

Jedes Element des Modells kann mit Attributen beschrieben werden. Dabei beschreibt ein Schlüssel das Attribut und dessen Inhalt. Ein Beispiel ist das Attribut arch für die Architektur eines IT-Systems.

Server1.arch="x86"

Server1.mem="2GB"

Server1.stor="100 GB"

beschreibt mit diesen Attributen einen Intel x86 kompatiblen Server mit seiner Hauptspeicher- und Festplattenausstattung. Die Bezeichner für die Attribute werden vom Modell nicht festgelegt, sie sind bei der Modellierung entsprechend zu wählen.

Die Abbildung 46 zeigt eine grafische Darstellung. Für einen Server sind hier seine Architektur, sein Speicherausbau und seine Plattenkapazität definiert. Für das dargestellte Speichersubsystem wurde die Anzahl der Festplatten und die Gesamtkapazität angegeben. Als Schema für die Benennung der Attribute bietet sich die Verwendung des Schemas des eingesetzten Managementsystems an. Die schon beschriebenen Tools IBM Tivoli, Microsoft MOM/SMS und andere Managementsysteme liefern an dieser Stelle eigene Schemata, in denen Informationen abgelegt werden. Meist basieren diese auf dem Common Information Model (CIM), einem Standard für entsprechende Schemata. Diese lassen sich hier sehr einfach integrieren.

Dynamische Attribute/Funktionen

Im Gegensatz zu statischen Attributen dienen dynamische Attribute und ihnen zugeordnete Funktionen dazu, sich ändernde Werte oder voneinander abhängige Attribute zu modellieren. Sie werden wie statische Attribute über die entsprechenden Funktionen einem Modellelement zugeordnet.

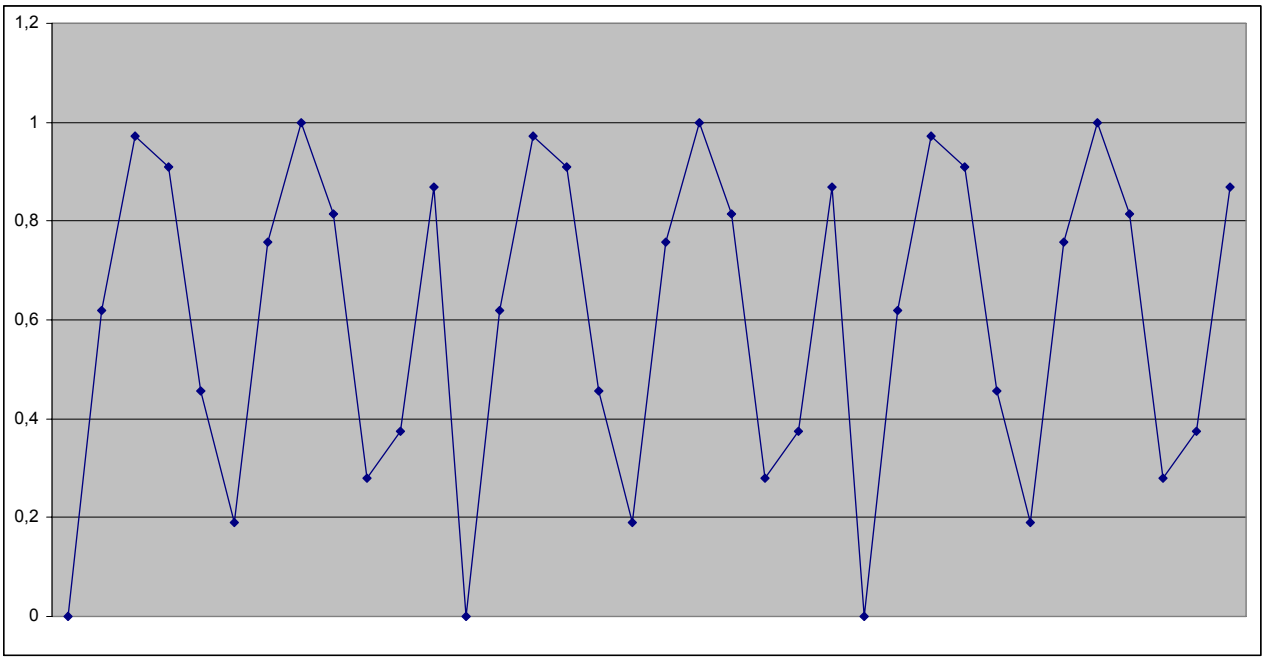

Abbildung 47 – Verlauf der Auslastung eines Servers

Abbildung 47 zeigt beispielhaft das Anforderungsprofil einer Anwendung an einen Server. Sind die Anforderungen an die Infrastruktur bekannt, so kann man diese, auf einen Referenzwert normiert[59], entsprechend modellieren.

functSAPrequiredbyTime(t) { abs(sin(3/t)) }

Die beschriebene Funktion ist in diesem Fall willkürlich. Meist wird sie sich nach den Anforderungen des Geschäftsprozesses richten. So ist es zum Beispiel möglich, dass ein Geschäftsprozess innerhalb eines gewissen Zeitfensters durchgeführt werden muss. Dieses Zeitfenster lässt sich dann durch eine entsprechende Funktion modellieren. Bei der Wahl der Bennennung der Funktion sollte, wie bereits beschrieben, ein einheitliches Namenssystem gewählt werden.

Wir lassen für die Funktionen keine Beeinflussung anderer Attribute oder den Zugriff auf andere Attribute zu. Alle Informationen zur Berechnung müssen innerhalb der Funktion oder aus den übergebenen Parametern errechnet werden. Dies verhindert Zirkelbezüge und unentscheidbare Funktionen innerhalb des Modells.

Attribute und Instanzen

Alle Attribute der Default-Instanz gelten auch für alle konkreten Instanzen. Für diese Instanzen können die Werte jedoch überschrieben werden. Dies realisiert das Informatik-Konzept der Verschattung. Wird ein Attribut auf einer konkreten Instanz nochmals definiert, so gilt der lokale Wert. Andererseits gelten Attribute, deren Wert nur in einer konkreten Instanz gesetzt sind, auch nur in dieser Instanz und sind für alle anderen Instanzen nicht gültig.

Funktionen auf Attributen

Für die Zuordnung von Attributen an Elemente definieren wir die notwendigen Funktionen. Die Funktionen sind für statische und dynamische Attribute identisch.

setAttrib („Attributschlüssel", „Attributwert"): Komponente x Wert → Komponente´

mit #Attribute Komponente + 1 = #Attribute Komponente `

getAttrib („Attributschlüssel"): Komponente → Attributwert

Formel 19 – Funktionen zum Umgang mit Attributen

5.1.7 Anforderungen

Anforderungen beschreiben Voraussetzungen, die erfüllt sein müssen, damit eine Komponente installiert und ausgeführt werden kann. Anforderungen wirken dabei auf die Attribute der Modellelemente. Sie beschreiben den betroffenen Schlüssel sowie die zu erfüllende Bedingung.

Anforderungen werden innerhalb des Attribut Req definiert. Anforderungen beschreiben Voraussetzungen, die erfüllt sein müssen.

Definition 26 – Anforderungen

Für die Beschreibung von Anforderungen sind logische Operatoren sowie <> erlaubt. Die Anforderung

SK1.REQ=(OS="Windows" und Version >"5.0")

Formel 20 – Syntax von Anforderungen

[59] Hierbei spielt die Wahl eines Referenzwertsystems eine wichtige Rolle. Geeignet sind hier Anwendungsbezogene Benchmarks wie die SAPs für die Leistungsfähigkeit bei SAP Anwendungen. Diese machen verschiedene Systeme und Architekturen miteinander vergleichbar. Für Datenbanken existieren vergleichbar Benchmarks, die auch einen Vergleich unterschiedlicher Datenbanksysteme erlauben.

für eine Softwarekomponente beschreibt zum Beispiel, dass mindestens Windows in einer Version größer 5.0 (WinXP/Win2003) eingesetzt werden muss,

(OS="Mac OS" und Version >="10.4")

entsprechend, dass mindestens Mac OS X Tiger installiert sein muss. Dabei sind in diesen Anforderungen auch andere Informationen versteckt. So haben Mac OS und Windows Anforderungen an die Hardware, auf denen sie ausgeführt werden können. Diese Anforderungen gelten dann auch für die jeweilige Softwarekomponente. So ergeben sich durch das Aufschlüsseln der Anforderungen einer Softwarekomponente neue Anforderungen, die implizit gefordert sind und für die jeweilige Softwarekomponente nicht modelliert werden müssen. Im Beispiel von Windows sind dies, eine Intel Architektur der Familie x86 oder x64. Wichtig sind die Anforderungen auch, um die Beziehungen im Softwaremodell zu beschreiben. Diese werden über Requirements modelliert, wir beschreiben dies ausführlich im Abschnitt 5.4. Auch dynamische Attribute lassen sich, wie bereits beschrieben, als Anforderungen definieren. So lassen sich auch Anforderungen definieren, die von anderen Attributen abhängen.

functCPUrequired { CPUrequired(KomponenteB)}

In diesem Fall beschreibt die Funktion, dass für sie dieselben Anforderungen gelten, wie für eine Komponente B. Diese Modellierungstechnik lässt sich in einem Tool dazu benutzen, abhängige Anforderungen auch auf die tatsächliche oder geplante Infrastruktur anzuwenden. Entsprechend modellierte Anforderungen können so helfen, das Entstehen von Engpässen durch die unterschiedliche Dimensionierung von voneinander abhängigen Komponenten in einem komplexen System zu verhindern.

Die Modellierung technischer Anforderungen ist nicht nur für die Ausführbarkeit von Softwarekomponenten wichtig. Servicelevelagreements beinhalten häufig auch Anforderungen an das Laufzeitverhalten eines Systems (Reaktionsgeschwindigkeit, Laufzeit von Berechunung, Verhalten bei Lastspitzen….). Diese Anforderungen, die meist auf der Ebene der Geschäftsprozesse definiert werden, müssen auf der technischen Ebene entsprechend implementiert werden. Dazu müssen die abstrakt definierten Anforderungen des Geschäftsprozesses entsprechend umgesetzt werden. Für die Rechenleistung der Systeme bieten sich in diesem Fall anwendungsnahe Wertschemata wie die SAP- oder SQL Benchmarks an, da diese die Leistungsfähigkeit eines IT-Systems deutlich besser beschreiben als einfachere Werte wie die Zahl der Rechenoperationen pro Zeiteinheit.

Bei der Modellierung der Anforderungen muss das für das Modell gewählte Namenschema für Attribute beachtet werden.

Funktionen auf Anforderrungen

Wir definieren Funktionen, mit denen sich Anforderungen an Komponenten modellieren lassen.

setReq („Schlüssel", „Wert"): Komponente x Wert → Komponente

getReq („Schlüssel"): Komponente → Wert

Formel 21 – Funktionen zum Umgang mit Anforderungen

Die entsprechenden Methoden finden sich auch wieder bei der Konsistenzprüfung des Modells.

5.1.8 Kanäle

Kanäle beschreiben Verbindungen zwischen verschiedenen IT-Systemen sowohl auf der physikalischen als auch auf der logischen Ebene und modellieren damit die gesamte Kommunikationsinfrastruktur eines IT-Dienstes.

Kanäle sind wie folgt definiert:

> *Kanäle beschreiben eine Verbindung zwischen zwei oder mehreren Endpunkten. An einen Kanal sind die Schnittstellen der verschiedenen Endpunkte gebunden, die über den Kanal miteinander verbunden sind. Diese Schnittstellen müssen zueinander kom-*

patibel sein. Kanäle werden in Hierarchien geordnet, wobei sich aus der Hierarchie eine Abhängigkeitsbeziehung ergibt.

Definition 27 – Kanal

Die Syntax von Kanälen ist gegeben durch:

Kanal = (Endpunkt, Endpunkt [{ ,Endpunkt }] : { Schnittstelle, Schnittstelle [{ ,Schnittstelle}*])*

mit #Endpunkt = #Schnittstelle

und $\forall$ *Sa* $\in$ *S gilt:* $\notin$ *Sn* $\in$ *S mit compat(Sa , Sn) = false*

Formel 22 – Syntax von Kanälen

Kanäle umfassen mindestens zwei Endpunkte und zwei Schnittstellen. Damit ist es möglich, auch mehr als zwei Teilnehmer eines physikalischen Netzes zu modellieren. Ein Beispiel hierfür wäre ein Netzwerk, das auf Broadcast Nachrichtenaustausch basiert, wie bei einer BNC Verkablung. Bei einer BNC Verkabelung werden alle Rechner direkt mit einem Kabel verbunden, es werden keine aktiven Netzwerkkomponenten zwischen den Servern verwendet, sondern es werden nur passive Stecker und so genannte Terminatoren an den beiden Enden der Verkabelung verwendet. Diese Netzwerkstruktur lässt sich aber auch durch eine virtuelle Netzwerkkomponente modellieren, mit der alle Teilnehmer des Netzes verbunden werden. Damit ist ein Ausfall des Netzes, zum Beispiel durch das Entfernen eines Terminators, einfacher analysierbar. Für andere Netztopologien wie Fibre channel Loops, also Ringnetze, ist die vereinfachte Form der Modellierung aber hilfreich. Die Art der Modellierung richtet sich vor allem nach dem Detaillierungsgrad und der gewünschten Abstraktion durch das Modell.

Kanäle sind durch ihre Schnittstellendefinitionen in Hierarchien eingeteilt. Dieses Konzept wird in der Informatik auch als Typisierung bezeichnet. Im Gegensatz zu einer einfachen Typisierung von Kanälen, bei der die Nachrichtentypen des Kanals festgelegt werden, ergeben durch die Hierarchien zusätzliche Abhängigkeiten der Kanäle voneinander. Die Formel verdeutlicht dies

(Ethernet:ip:tcp:http)

Kanal1 = (Client, Server : (Ethernet[00:00:00:00:00:01] , Ethernet[00:00:00:00:00:02]))

Kanal2 = (Client, Server : (Ethernet[00:00:00:00:00:01]:IP[192.168.0.1] , Ethernet[00:00:00:00:00:02]: IP[192.168.0.2]))

Kanal3 = (Client, Server : (Ethernet[00:00:00:00:00:01]:IP[192.168.0.1]:tcp , Ethernet[00:00:00:00:00:02]: IP[192.168.0.2]:tcp))

Kanal4 = (Client, Server : (Ethernet[00:00:00:00:00:01]:IP[192.168.0.1]:tcp:http , Ethernet[00:00:00:00:00:02]: IP[192.168.0.2]:tcp:http))

Formel 23 – Kanäle für http

Die Kanäle werden dabei von Kanal 1, der das physikalische Netzwerkkabel zwischen den beiden Servern beschreibt über die Kanäle 2 und 3 bis hin zur Verbindung auf http Ebene, die Kanal 4 modelliert, verfeinert. Die Modellierung der verschiedenen Verfeinerungsstufen kann dabei in einem Tool automatisch erfolgen.

Die Kanäle können auch mehr als zwei Hierarchieebenen umspannen. Wie auch die Schnittstellen verbinden die Kanäle das Softwaremodell und das Infrastrukturmodell. Kanäle können darüber hinaus auf mehr als einem anderen Kanal basieren. Wir gehen im Verlauf dieses Abschnitts noch auf die Modellierung entsprechender Kanäle ein.

Die Notwendigkeit für die explizite Modellierung von Kanalhierarchien ergibt sich aus den erweiterten Möglichkeiten verschiedener Protokollstacks. Es muss nicht nur die physikalische Ebene des Netzes modellierbar sein, sondern auch höhere Ebenen der Kommunikation. Verschiedene

Protokollstacks erlauben es, dass sich auf verschiedenen Schnittstellenebenen logische Einheiten und Grenzen in die jeweiligen Netze einführen lassen. Dies lässt sich am Beispiel von TCP/IP verdeutlichen. IP Netze lassen sich auf Softwareebene partitionieren. Die Protokollfamilie sieht dafür das Konzept der Subnetze vor. Ein Rechner mit der IP Adresse 192.168.0.1 in einem Klasse C Subnetz (255.255.255.0) kann einen Rechner mit der IP Adresse 192.168.1.1 nicht ohne einen entsprechenden Router erreichen, auch wenn beide über einen direkte physikalische Verbindung zueinander verfügen. Würde nur diese physikalische Verbindung modelliert, wären die Einschränkungen durch die Protokolle nicht aus dem Kanal ersichtlich. Gleiches gilt zum Beispiel auch beim Einsatz von VLAN's in Ethernetnetzwerken. Das Konzept tritt noch deutlicher bei virtuellen Infrastrukturen zutage, bei denen selbst die physikalischen Infrastrukturkomponenten der virtuellen Maschine nicht wirklich existieren, sondern emuliert werden. Diese müssen auf die tatsächlichen physikalischen Komponenten und Kanäle des Hosts abgebildet werden.

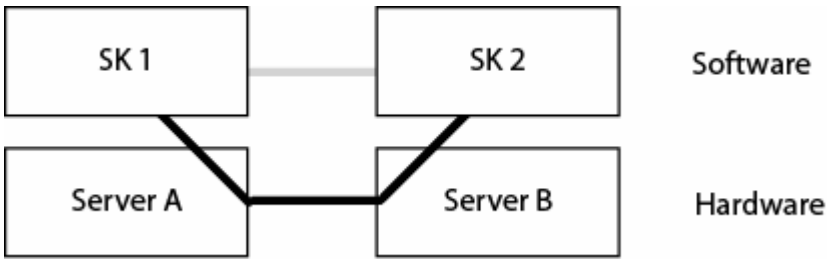

Abbildung 48 – Kanäle auf den verschiedenen Ebenen

Für die Verwendung unseres Modells ist die Definition von Schnittstellenhierarchien notwendig, die bereits implizit die Hierarchien der Kanäle definieren. Wie beschrieben ergeben sich diese Hierarchien bei direkten Verbindungen von zwei Rechnern implizit. Hierarchien müssen aber auch explizit modelliert werden können. Werden in einem zu modellierenden Szenario mehrere Rechner über Infrastrukturkomponenten verbunden, so sind nicht mehr alle Schnittstellenhierarchien auf allen IT-Systemen verfügbar. Teilweise basieren hier Kanäle auf mehreren Kanälen einer untergeordneten Hierarchie. Wir verdeutlichen dies in einem der nächsten Abschnitte.

Wir lassen für Kanäle keine Instanzen zu. Diese Einschränkung trägt zur Eindeutigkeit der Kanäle bei und schränkt die Modellierung nicht ein, da Kanäle durch ihre Hierarchie und ihre Endpunktbeziehungen nur schwer Verallgemeinert werden können[60].

Funktionen auf Kanälen

Wir definieren einige Funktionen auf Kanälen, die Kanäle zwischen verschiedenen IT-Systemen definieren und die Hierarchien der Kanäle festlegen.

$$\{ Endpunkt \}^n \times \{ Schnittstellenhierarchie \}^n \rightarrow Kanal \text{ mit } n>1$$

$$createKanal (E \{Endpunkt\}^n, S \{Schnittstellenhierarchie\}^n): \rightarrow KANAL$$

$$mit \ \forall S_a \in S \text{ gilt: } \notin S_n \in S \text{ mit } compat(S_a , S_n) = false$$

$$addEndpunkt (E, K) : Komponente, Kanal \rightarrow Kanal$$

$$mit \ \forall S_a \in S \text{ gilt: } \notin S_n \in S \text{ mit } compat(S_a , S_n) = false$$

Formel 24 – Funktionen auf Kanälen

Die Funktion createKanal erzeugt einen Kanal zwischen zwei oder mehreren Endpunkten. Die Funktion addEndpunkt fügt einem bereits existierenden Kanal einen weiteren Endpunkt hinzu.

[60] Ein mögliches Beispiel wäre, alle Ethernet-Verbindungen eines Modells adressieren zu können. Dies ist aber auch durch die Kanalhierarchien möglich.

Dies entspricht auf einer technischen Ebene zum Beispiel dem Anschluss eines weiteren IT-Systems an ein Ringnetzwerk.

Kanäle sind in Hierarchien angeordnet, für die wir die entsprechenden Funktionen definieren.

CreateKanal (ServerA[0], ServerB[0], Layer1[0], Layer1[0])= K1

CreateKanal (ServerA[0], ServerB[0], Layer1[0]:Layer2[0], Layer1[0]:Layer2[0])=K2

Kanal → Kanal

parent(K2) = K1

Kanal → { Kanal } *

children(K1) = K2

Kanal x Kanal → Bool

ischild (K2, K1) true, wenn gilt: K2 $\in$ *children (K1)*

ischild (K1, K2) false, wenn gilt: K1 $\notin$ *children (K2)*

isparent (K2, K1) true, wenn gilt: K2 $\in$ *children (K1)*

isparent (K1, K2) false, wenn gilt: K1 $\notin$ *children (K2)*

Kanal x Kanal → Kanalhierarchie

addsubKanal(Kanal1, Kanal2)=Kanalhierarchie 1

Formel 25 – Funktionen für Kanalhierarchien

Die Funktion isparent und ischild dienen zur Überprüfung der Abhängigkeit von zwei Kanälen voneinander. Die Funktion children gibt alle Kanäle an, die auf dem angegebenen Kanal basieren. Die Funktion parent liefert den Kanal, auf dem der übergebene Kanal basiert.

Erweitertes Beispiel

Kanäle sind eines der Grundelemente unseres Modells. Wir führen deshalb noch ein zweites Beispiel ein, das erweiterte Konzepte verdeutlicht. Zwei Server sind über einen Ethernet Switch miteinander verbunden. Wir vereinfachen für unser Beispiel die Schnittstellenhierarchie auf die beiden Hierarchieebenen Ethernet und TCP/IP.

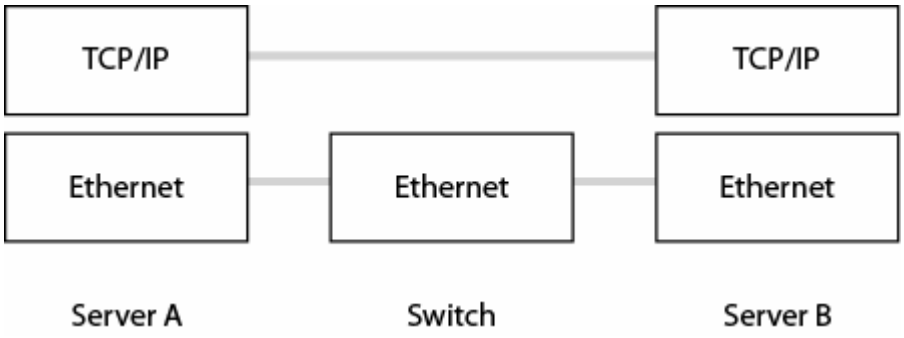

Abbildung 49 – Kanäle zwischen IT-Systemen und IT-Infrastrukturkomponenten

Wie aus der Abbildung 49 deutlich wird, existiert ein Kanal auf der Ebene Ethernet zwischen dem Server A und dem Layer 2 Switch, sowie zwischen dem Layer 2 Switch und dem Server B. Dies entspricht der Verbindung zwischen der jeweiligen Netzwerkkarte des Servers und den Ports des Switches. Die beiden Server haben einen Kanal auf der TCP/IP Ebene, der sie miteinander verbindet. Dieser stützt sich auf der physikalischen Verbindung, die darunter liegt ab.

Entsprechend wird diese Beziehung in den Hierarchien der Kanäle gespeichert. Die Funktionen illustrieren dieses konkrete Beispiel:

CreateKanal (ServerA, Switch, Ethernet, Ethernet)= K1

CreateKanal (ServerB, Switch, Ethernet, Ethernet)= K2

CreateKanal (ServerA, ServerB, Ethernet:TCP/IP, Ethernet:TCP/IP) = K3

addsubKanal(K3, K2)=KH1

addsubKanal(K3, K1)=KH2

Formel 26 – Funktionen für zwei Server über Switch

Werden zur Steigerung der Ausfallsicherheit zwischen zwei Systemen mehrfache physikalische Netzwerkverbindungen eingesetzt und diese zu einer logischen zusammengefasst, wie bei Intel Switch Fault Tolerance oder IEEE 802.3ad, so werden die höheren Kanalhierarchien, die sich auf diese Verbindung abstützen, auch auf beide physikalische Verbindungen in der Kanalhierarchie abgebildet.

5.1.9 Zusammenfassung

Wir haben die übergreifenden Elemente, die sich auf mehrere oder alle Teilmodelle beziehen, im diesem Abschnitt beschrieben. In den nächsten Abschnitten definieren wir unsere vier Teilmodelle und stellen die Funktionen, die diese erzeugen, vor.

5.2 Funktionsmodell

Das Funktionsmodell beschreibt auf einer abstrakten Ebene die Features, die durch das Geschäftsprozessmodell, das Softwaremodell und das Infrastrukturmodell realisiert werden. Wir verwenden hier den Begriff Features und nicht Funktion, da wir den Begriff Funktionen im Modell bereits für die Funktionen zur Erstellung des Modells verwenden. Um Verwechslungen zu vermeiden, verwenden wir für die Funktionshierarchien den Begriff Features.

Wir haben bereits im vorigen Kapitel beschrieben, wie das Funktionsmodell die Kommunikation zwischen dem Entwickler und dem Geschäftsprozessverantwortlichen verbessern kann, indem die Abbildung der Softwarekomponenten auf die Geschäftsprozesse auf der Basis der realisierten Features erfolgt. Damit wird die Gemeinsamkeit der Elemente der beiden Ebenen durch das Feature beschrieben. Darüber hinaus dient das Funktionsmodell auch zur besseren Strukturierung unserer Modelle. Wir beschreiben damit auf einer abstrakten Ebene die verschiedenen Features, die die unterschiedlichen Modelle verbinden. So lässt sich damit zum Beispiel auch prüfen, welcher Server die Funktion Datenhaltung realisiert. Es erschließt sich eine zusätzliche Sicht auf die IT-Dienste. Aus dieser Perspektive ist auch die Frage beantwortbar, ob ein Feature durch mehrere Systeme realisiert wird. Ist dies der Fall, kann analysiert werden, inwieweit die beiden Realisierungen notwendig sind und falls dies zutrifft, trotzdem an einer Standardisierung gearbeitet werden. Darüber hinaus lassen sich so auch Vorgaben modellieren, wie entsprechende Features auf den jeweiligen Ebenen umgesetzt werden sollen, wenn dafür Platzhalterobjekte eingeführt werden, die die Musterimplementierung darstellen.

Das Funktionsmodell beschreibt Features, die in Hierarchien angeordnet werden können. Es beschreibt auf Basis dieser Features die Beziehungen zwischen den Ebenen und stellt damit eine Abstraktion der konkret implementierten Funktionen dar. Würde diese Modellierung innerhalb der Beziehungen stattfinden und die Beziehung entsprechend um ein Attribut angereichert, wäre ein Vergleich unterschiedlicher Realisierungen nicht mehr möglich.

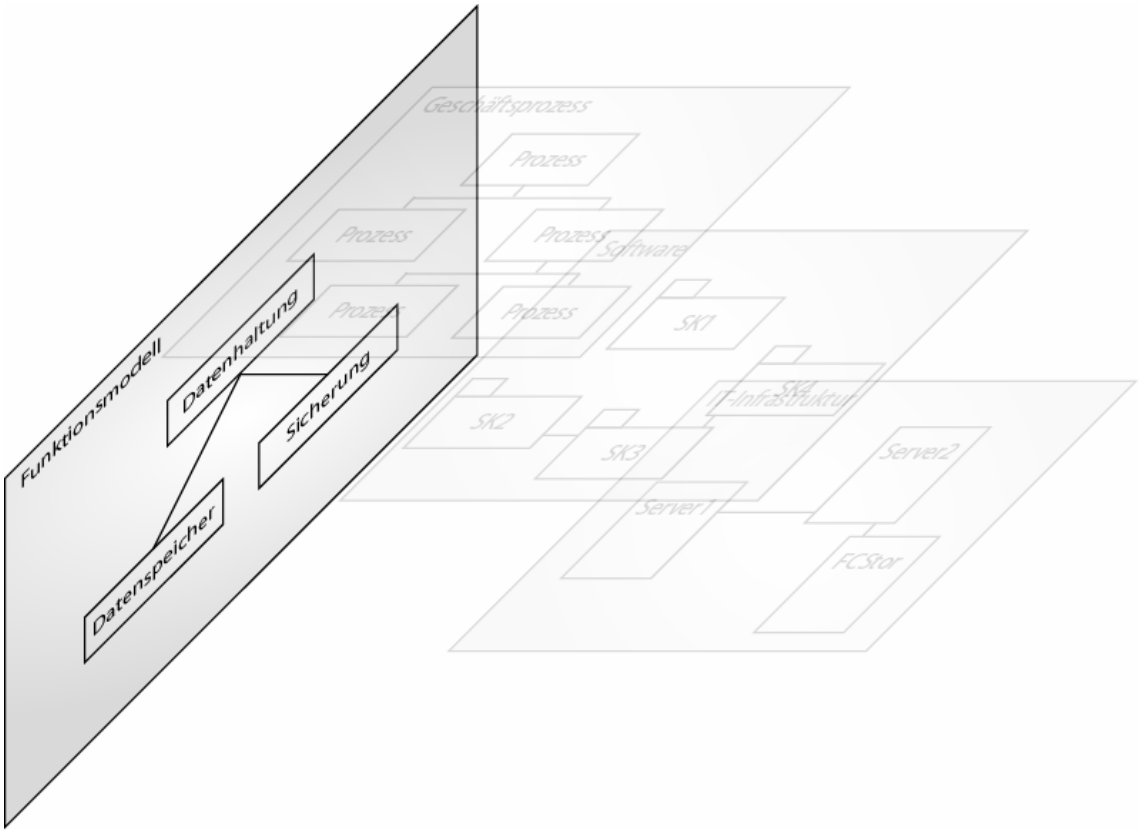

Abbildung 50 – Das Funktionsmodell

5.2.1 Definition

Das Funktionsmodell ist aufgebaut aus Features, die in Hierarchien angeordnet werden können. Wir definieren ein Feature als:

> *Ein Feature beschreibt eine Funktion oder Aufgabe, die von einer oder mehreren Objekten in einem oder mehreren Teilmodellen erbracht wird. Das Feature wird dabei durch einen Bezeichner beschrieben. Der Bezeichner kann dabei frei gewählt werden, muss innerhalb des Funktionsmodells aber eindeutig sein. Features können in Hierarchien angeordnet und so strukturiert werden.*

Definition 28 – Feature

Die Syntax von Features ist gegeben durch:

*Feature = (Bezeichner) { [Instanz] } **

Formel 27 – Syntax von Features

Die Hierarchien von Features ermöglichen die Modellierung von Verfeinerungen von einzelnen Features und damit eine Strukturierung des Funktionsmodells. Die Abbildung 51 beschreibt ein Beispiel für eine Featurehierarchie. Für die Datenhaltung werden hier die beiden Feature Datenspeicher und Sicherung definiert. Auf diese beiden Features werden dann die entsprechenden Softwarekomponenten und die IT-Systeme, die dazu benutzt werden, abgebildet.

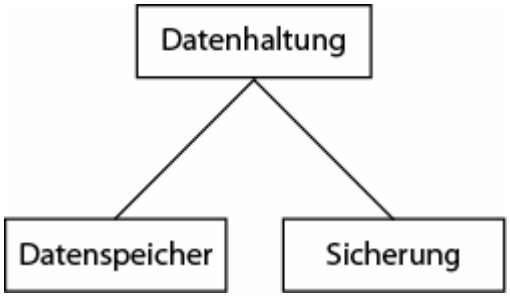

Abbildung 51 – Featurehierachie

5.2.2 Instanzen im Funktionsmodell

Wir lassen Instanzen im Funktionsmodell zu, empfehlen den Einsatz jedoch nur in sehr komplexen Modellen. Es besteht die Gefahr, dass durch Instanzen die Instanzen der einzelnen Teilmodelle nachgebaut werden. Ein Beispiel ist hier die Speicherung von Logbüchern. Diese können in Dateien oder in Datenbanken abgelegt werden. Wird für jede vorhandene Implementierung eine Instanz des Features Logbuch modelliert, so ist für diese Feature zwar eine Trennung der untergeordneten Features je nach Implementierung möglich, hier werden aber nur die Funktionen der verschienen Implementierungen beschrieben. Wichtig für das Funktionsmodell ist aber gerade die Abstraktion von diesen Implementierungsdetails und den anderen Teilmodellen, da hier ein Überblick über die unterschiedlichen Realisierungen ermöglicht und gleiche Realisierungen vergleichbar gemacht werden sollen.

Ein Beispiel, in dem die Modellierung von Instanzen im Funktionsmodell sinnvolle wäre, ist ein Szenario, in dem sehr viele Webapplikationen verschiedene Autorisierungstechniken verwenden, die alle dasselbe Feature (Autorisierung) mit unterschiedlichen Mitteln anbieten. Der Microsoft Internet Information Server bietet hier bereits drei verschiedene Techniken an, ähnliches gilt für andere Webserver. Hier wäre es sinnvoll, die verschiedenen Autorisierungsarten als Instanzen zu modellieren. Dieselbe Funktionalität lässt sich aber auch durch die Modellierung von Featurehierarchien erreichen. Hier muss bei der Modellierung abgewägt werden, wie detailliert oder wie kompakt das Funktionsmodell ausfallen soll.

5.2.3 Grafische Darstellung

Wir verwenden Kästen für die grafische Darstellung der Features und Linien, um die Hierarchiebeziehungen abzubilden. Die Kanten sind dabei nicht gerichtet. Die Hierarchie wird durch die vertikale Ausrichtung der Objekte erreicht. Die Abbildung 51 verdeutlicht dies.

Unsere Funktionsmodelle sind im Sinne der Modellierungstechnik Wälder, weil mehrere Funktionsbäume unabhängig voneinander in einem Funktionsmodell existieren können.

5.2.4 Abbildungen zwischen den Teilmodellen

Die Abbildung zwischen den Teilmodellen erfolgt durch Funktionen, die wird bei den verschiedenen Modellen im weiteren Verlauf angeben. Diese Verbindungen sind ungerichtet, da die Abbildung in beide Richtungen gilt und keine Hierarchie zwischen den Ebenen vorhanden ist.

Wichtig ist die Abbildung vor allem zwischen dem Softwaremodell und dem Geschäftsprozessmodell. Die Abbildung zwischen dem Funktionsmodell und dem Infrastrukturmodell ergibt sich durch das Deployment von Software auf den verschiedenen IT-Systemen.

5.2.5 Funktionen auf Funktionsmodellen

In diesem Anschnitt definieren wir die Funktionen, die in unserem Funktionsmodell Features erstellen und Operationen auf ihnen ermöglichen. Damit werden Features in das Funktionsmodell eingefügt und die Beziehungen zu den anderen Teilmodellen beschrieben.

Anlegen eines Features

Die Funktion zum Anlegen eines Features definiert ein Featureobjekt im Funktionsmodell. Dabei kann die Instanz mit angegeben werden. Wird die Instanz nicht angegeben, wird das Objekt mit der Defaultinstanz erstellt.

Bezeichner [x Instanzbezeichner] → Feature

createF(„Bezeichner" [, „Instanzbezeichner"]) = F

createF(„Datenhaltung") = F1, dabei gilt:

Sei F die Menge aller Features. $\forall F_a \in F$: *Bezeichner*(F_a) $\neq$ *Bezeichner(F1)*

Formel 28 – Funktion zum Anlegen eines Features

Die Formel definiert ein Feature Datenhaltung, wie in Abbildung 51 beschrieben.

Hierarchische Anordnung

Wir definieren zur besseren Strukturierung unserer Modelle Hierarchien in den einzelnen Modellen. Wir definieren dazu die Verfeinerung und die Abstraktion von Features, wie bereits bei den Grundkonzepten unseres Modells beschrieben.

Verfeinerung

Die Verfeinerung von Features ermöglicht die Unterteilung von Features in unterschiedliche Teilfeatures. Features werden dabei in Hierarchien eingeordnet und weiter unterteilt. Die Abbildung 51 verdeutlicht dies.

createF(„Datenspeicher)= Fs1

Feature x Feature → Feature x Feature x Hierarchiebeziehung

addsubF(F1, Fs1)

Formel 29 – Funktionen zur Verfeinerung eines Features

Die Funktion addsubF ordnet zwei existierende Features einander in einer entsprechenden Beziehung zu. Die Hierarchien sind nicht exklusiv.

Wir definieren die Funktion children auf Hierarchien, die alle in der Featurehierarchie untergeordneten Features ausgibt, bzw. die Funktion parent, die alle übergeordneten Features auflöst:

Feature → { Feature } *

parent(Fs1) = { F1 }

children(F1) = { Fs1 }

Formel 30 – Kindelemente einer Hierarchiebeziehung

Wir definieren Funktionen zur Überprüfung von Hierarchiebeziehungen:

Feature x Feature → Bool

ischild(Fs1, F1) = true, wenn gilt: Fs1 $\in$ *children (F1)*

ischild(F1, Fs1) = false, wenn gilt: F1 $\notin$ *children (Fs1)*

isparent(Fs1, F1) = false, wenn gilt: F1 $\notin$ *children (Fs1)*

isparent(F1, Fs1) = true, wenn gilt: Fs1 $\in$ *children (F1)*

isroot(F1) = true, wenn gilt:

Sei F die Menge aller Features

$\forall$ $F_a \in$ *F gilt ischild(F1,* F_a*)=false und*

E $F_b \in$ *F mit ischild(*F_b*, F1)=true*

isatomar(F1) = true, wenn gilt:

Sei F die Menge aller Features

$\forall$ $F_a \in$ *F gilt isparent(*F_a *,F1)=false*

Formel 31 – Funktionen auf Hierarchiebeziehungen

Die Funktion ischild dient dabei zur Überprüfung, ob ein Feature in der Hierarchie eines anderen Features vorkommt. Die Funktion isroot gibt an, ob ein Feature ein Wurzelelement in einer Featurehierachie ist. Die Funktion isatomar gibt an, ob ein Feature ein Blattelement in einer Featurehierachie oder für sich allein stehend ist.

Abstraktion

Das Gegenstück zur Verfeinerung von Features ist die Abstraktion. Bei diesem Vorgang werden zwei Features zu einem Feature zusammengefasst[61]. Bei der Abstraktion könne dabei Informationen gelöscht und verloren werden.

Feature [x Feature] x Feature → Feature

abstractF(F1, F2) → F2, falls isatomar(F1) = true und isparent(F2,F1)=true;

Formel 32 – Funktion zur Abstraktion von Features

Wir lassen die Abstraktion dabei nur von atomaren Features zu. Wir führen bei den weiteren Modellebenen Funktionen ein, die andere Modelle mit Features verbinden. Wird ein Feature abstrahiert, das mit anderen Objekten verbunden war, werden diese mit dem Feature verbunden, zu dem abstrahiert wurde.

[61] Eine Spezialform der Abstraktion ist die Realisierung einer Zoom-Funktion in einem Tool. Damit wird nicht das Modell an sich, sondern die Darstellung geändert. Die Funktionalität entspricht dabei unserer Abstraktionsfunktion, die Informationen gehen in diesem Fall aber nicht verloren, da nur die Darstellung und nicht das Modell an sich verändert wird.

5.3 Geschäftsprozessmodell

Die Geschäftsprozessebene wird beschrieben durch unser Geschäftsprozessmodell, das die tatsächlich realisierten Geschäftsprozesse und ihre Abläufe beschreibt. Auf dieser Ebene existieren bereits verschiedene Modellierungstechniken (Konzept Veronika Thurner, ARIS), die weit über unsere Anforderungen hinausreichen. Sie ermöglichen neben der Modellierung von IT-System-basierten Geschäftsprozessen auch die Beschreibung allgemeiner Geschäftsprozesse. Diese zeichnen sich zum Beispiel durch menschliche Akteure aus.

Für unser Modell wichtig ist die Modellierung von Geschäftsprozessen, die auf IT-Systemen ausgeführt werden. Existiert ein Modell, das weitere Informationen erhält, ist es integrierbar, die Modellierung zum Beispiel menschlicher Akteure ist aber nicht notwendig. Wir werden uns an dem von Thurner [Thu04] vorgestellten Modell orientieren und anhand dieses Modells unsere Modellelemente definieren. Eine Abbildung unserer Geschäftsprozesse in ein ARIS Modell ist im Kapitel 6.4 angegeben.

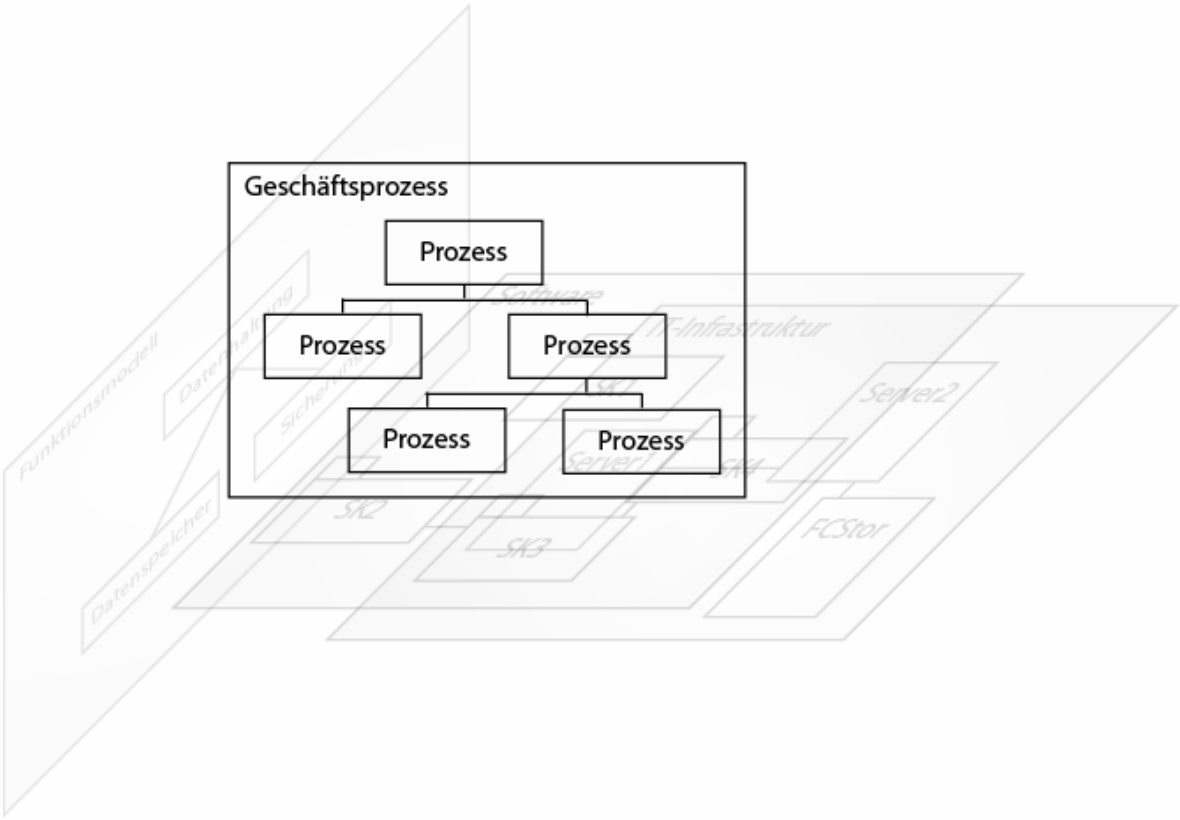

Abbildung 52 – Die Ebene der Geschäftsprozesse

Thurner [Thu04] definiert Geschäftsprozesse, Geschäftsprozessnetze und Geschäftsprozessmodelle. Geschäftsprozesse beschreiben dabei einzelne Prozesse, die für sich isoliert ausgeführt werden und nicht miteinander verbunden sind. Die Beziehungen zwischen den Geschäftsprozessen werden mit Kanälen[62] modelliert, die gerichtet sind und damit die Abläufe zwischen den verschiedenen Geschäftsprozessen darstellen. Aus den Geschäftsprozessen und ihren Kanälen ergeben sich Geschäftsprozessnetze. Diese stellen komplexe Abläufe aufbauend auf Geschäftsprozessen dar und können selbst wieder als Geschäftsprozesse aufgefasst werden, wenn ihre Detailprozesse verschattet werden. Die entsprechenden Informationen werden in Abstraktionsbeziehungen hinterlegt. Diese realisieren damit die Glass- und die Black-Box Sicht auf die jeweiligen Prozesse. Geschäftsprozesse werden zu Geschäftsprozessmodellen zu-

[62] Die Kanäle entsprechen nicht unseren Kanälen

sammengefasst. Das Geschäftsprozessmodell umfasst dabei alle Geschäftsprozesse und Geschäftsprozessnetze, sowie die die verschiedenen Abstraktionsbeziehungen und die Kanäle zwischen den Geschäftsprozessen.

In unserem Geschäftsprozessmodell beschreiben wir Geschäftsprozesse, sowie die Beziehung verschiedener Geschäftsprozesse untereinander. Wir stützen uns hier auf den Geschäftsprozessmodell von Thurner ab. Das von uns verwendete Geschäftsprozessmodell beinhaltet darüber hinaus die Abläufe zwischen den Geschäftsprozessen. Das Konzept der Black- und Glass-Box Sichten auf Geschäftsprozesse von Thurner findet sich in unserem Modell in den Hierarchien von Geschäftsprozessen.

Die im Modell von Thurner vorhandenen einzelnen Aktionen der Geschäftsprozesse sind für unser Modell nicht notwendig. Eine feingranulare Modellierung aller Abläufe eines Geschäftsprozesses auf die Softwarekomponenten führt zu einem Modell, das nicht mehr handhabbar ist. Darüber hinaus sind sowohl die Softwaresysteme als auch die IT-Systeme deutlich größer, als dass sie nur einen Ablauf eines Geschäftsprozesses implementieren.

Ein nicht behandelter und für uns wichtiger Aspekt ist in diesem Zusammenhang die Frage, inwieweit Instanzen von Geschäftsprozessen existieren. Werden IT-Dienste durch einen Hosting-Partner erbracht, so ist es möglich, dass identische Geschäftsprozesse für unterschiedliche Kunden erbracht werden. Ein Beispiel ist die Erfassung von Kundenaufträgen oder die Abrechnung mit Kreditkartenunternehmen, die dann jeweils für eine Kartenfirma an beide Kunden zur Verfügung gestellt werden. Im Modell von Thurner wäre dazu die Modellierung von zwei identischen Geschäftsprozessen möglich. Wir führen dazu das Konzept der Instanzen von Geschäftsprozessen ein und ermöglichen so auch eine einfachere Überprüfung der Abbildung identischer Geschäftsprozesse auf die jeweiligen Softwarekomponenten. Darüber hinaus ist damit eine Abstraktion über die Instanzen möglich.

5.3.1 Definition

Thurner [Thu04] definiert einen Geschäftsprozess als „ein Muster für einen Arbeitsablauf in einem System“. Wir definieren einen Geschäftsprozess als:

> *Ein Geschäftsprozess beschreibt einen Arbeitsschritt auf einer fachlichen Ebene. Ein Geschäftsprozess kann in einer Hierarchie verfeinert werden. Die Abläufe zwischen verschiedenen Geschäftsprozessen werden durch Abhängigkeiten modelliert*

Definition 29 – Geschäftsprozess

Die Syntax ist gegeben durch:

$$Geschäftsprozess = (Bezeichner)\ \{\ [Instanz]\ \}^{*}$$

Formel 33 – Syntax von Geschäftsprozessen

Die Wahl der Bezeichner bleibt wie auf den anderen Ebenen frei wählbar.

5.3.2 Instanzen auf der Geschäftsprozessebene

Wir führen neben einem Bezeichner für einen Geschäftsprozess, der diesen beschreibt, zusätzlich das Konzept der Instanzen ein. Durch Instanzen ist es möglich, Geschäftsprozesse, die mehrmals in einem Gesamtmodell vorhanden sind, aber auf unterschiedliche andere Ebenen zugeordnet werden sollen, zu modellieren. Damit wird die Modellierung von standardisierten Prozessen auf der Ebene der Geschäftsprozesse unterstützt, die durch verschiedene Softwaresysteme realisiert sind. Ein Beispiel sind Geschäftsprozesse wie Webshops eines Internetdienstleisters, die dieser seinen Kunden anbietet. Dabei wird ein komplexer Geschäftsprozess an den Dienstleister ausgelagert. Dieser Geschäftsprozess gliedert sich in verschiedene Unterprozesse, zum Beispiel „Rechnung schicken“, der für eine Vielzahl an Kunden erbracht wird. Der Dienstleister kann so dieselben Prozesse für die verschiedenen Kunden durch die jeweiligen Instanzen modellieren. Damit können verschiedene Geschäftsprozesse modelliert werden, die auf standardisierten Teilprozessen basieren.

Wichtig ist in diesem Zusammenhang nicht nur die Unterscheidung der verschiedenen Kunden, sondern auch die Möglichkeit, diesen Prozess auf verschiedene Softwarekomponenten abzubilden. So lassen sich für verschiedene Kunden verschiedene technische Implementierungen modellieren, zum Beispiel eine Implementierung für Kleinunternehmen und eine Implementierung für Mittelständler. Die Attribute und Eigenschaften der Defaultinstanz des Geschäftsprozesses teilen dabei alle Instanzen des Geschäftsprozesses.

5.3.3 Attribute auf der Geschäftsprozessebene

Geschäftsprozesse können durch das bereits beschriebene Mittel der Attribute um weitere Informationen angereichert werden. Im Bezug auf das Anwendungsszenario unseres Modells sind hier vor allem Anforderungen an die Ausführung des Geschäftsprozesses wichtig. Dies kann Reaktionszeiten, Ausführungszeiten und andere Informationen betreffen, die sich auch in den SLAs des jeweiligen Dienstes finden. Darüber hinaus ist es sinnvoll, hier Informationen über Sicherheitseinschränkungen eines Prozesses zu modellieren.

Werden Informationen über den erwirtschafteten Wert eines Dienstes hinterlegt, so kann das Modell als Hilfe dienen, Ausfälle und Einschränkungen in der Ausführung der IT-Dienste monetär zu bewerten. Einen Ansatz dazu stellen wir unter 9.4 vor.

5.3.4 Abläufe

Wir definieren und verwenden auf der Ebene der Geschäftsprozesse im Gegensatz zu Thurner nur das Element Geschäftsprozess und verzichten auf die Modellierung der einzelnen Aktionen. Wichtig ist die Möglichkeit, Geschäftsprozesse hierarchisch zu untergliedern und Abläufe zwischen Geschäftsprozessen darstellen zu können.

Thurner verwendet dazu das Konzept der Geschäftsprozessnetze. Wir bilden die Abläufe durch Beziehungen zwischen den Geschäftsprozessen ab. Diese Beziehungen sind dabei gerichtet und werden durch Geschäftsprozessabläufe modelliert. Die einzelnen Abläufe gelten dabei für alle Instanzen eines Geschäftsprozesses[63]. Dabei können verschiedene Ausprägungen innerhalb eines Ablaufes existieren. So können bei zwei Geschäftsprozessen, die zu einem Ablauf verbunden sind, von einem Geschäftsprozess nur eine Instanz existieren, die für alle Kunden gleich ist, für den zweiten Geschäftsprozess jedoch für jeden Kunden eine Instanz. Wichtig ist hier, die eindeutige Wahl der Instanzbezeichner durch das Modell.

Ein Geschäftsprozessablauf beschreibt eine Beziehung zwischen zwei Geschäftsprozessen, bei denen ein Geschäftsprozess von einem anderen abhängig ist.

Definition 30 – Geschäftsprozessablauf

Die Syntax ist gegeben durch:

Geschäftsprozessablauf = Geschäftsprozess x Geschäftsprozess

Formel 34 – Syntax eines Geschäftsprozessablaufs

Neben den Prozessabläufen definieren wir Hierarchien zwischen den Geschäftsprozessen. Dies entspricht dem Konzept der Glass-Box und Black-Box Sicht von Thurner. Die Hierarchien dienen vor allem der Strukturierung der Geschäftsprozesse. Auch hier definieren wir, dass alle Instanzen eines Geschäftsprozesses in die Hierarchiebeziehung eingebunden werden.

Geschäftsprozesse lassen sich in Hierarchie einordnen. Diese Hierarchiebeziehungen dienen der Strukturierung des Geschäftsprozessmodells

Definition 31 – Geschäftsprozesshierarchie

[63] Wir gehen davon aus, dass Abläufe, bei denen die verschiedenen Instanzen eines Geschäftsprozesses in unterschiedliche Abläufe eingebettet wären, diese so unterschiedlich sind, dass sie in unterschiedliche Geschäftsprozesse modelliert werden sollten.

Die Syntax ist gegeben durch:

Hierarchiebeziehung = Geschäftsprozess x Geschäftsprozess

Formel 35 – Syntax einer Geschäftsprozesshierarchie

Eine konkrete Instanz unseres Geschäftsprozessmodells beschreibt komplexe IT-Dienste und IT-Infrastrukturen. Deswegen können auf der Ebene der Geschäftsprozesse verschiedene Geschäftsprozesse mit ihren jeweiligen Hierarchien existieren, die miteinander weder in einer Ablaufs- noch in einer Hierarchiebeziehung stehen. Die Gemeinsamkeit ergibt sich dann aus der Nutzung derselben IT-Infrastruktur oder derselben Softwarekomponenten. Grafisch entspricht die Darstellung dieser Geschäftsprozesse dann einem Wald.

5.3.5 Grafische Darstellung

Grafisch stellen wir Geschäftsprozesse durch einzelne Kästen dar. Hierarchien werden durch einfache Linien gekennzeichnet[64]. Abbildung 53 zeigt einen Geschäftsprozess (Prozess A) mit zwei Unterprozesse (Unterprozess 1 und 2). Durch die vertikale Anordnung wird die Geschäftsprozesshierarchie dargestellt.

Die beiden Unterprozesse werden nacheinander ausgeführt und stellen somit einen Geschäftsprozessablauf dar. Diese Abläufe zwischen den Geschäftsprozessen werden durch Pfeile verdeutlicht. Instanzen des Geschäftsprozesses werden in der grafischen Darstellung in eckigen Klammern angegeben, bei der Default Instanz kann diese Kennzeichnung zur Vereinfachung entfallen.

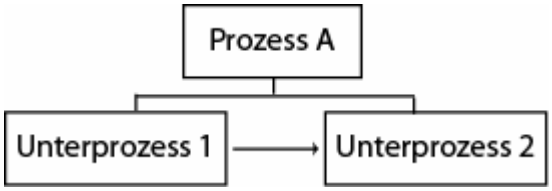

Abbildung 53 – Grafische Darstellung einer Geschäftsprozesshierarchie

5.3.6 Elemente eines Geschäftsprozesses

Die Geschäftsprozessebene unseres Modells enthält folgende Objekte und Attribute:

Geschäftsprozessobjekte beschreiben Geschäftsprozesse. Die Objekte beinhalten zur genaueren Beschreibung:

- Bezeichner
  Jeder Geschäftsprozess kann durch einen eindeutigen Bezeichner im Modell eindeutig referenziert werden. Die Wahl des Bezeichners ist dem Modellierer überlassen[65].
- Instanzbezeichner
  Jeder Geschäftsprozess wird mit einer Instanzbezeichnung im Modell angelegt. Wird diese nicht direkt angegeben, so wird der Geschäftsprozess als Standardinstanz angelegt[66]. Jede Instanz eines Geschäftsprozesses kann nur einmal im Modell existieren.

[64] Die Hierarchien unserer Geschäftsprozesse sind gerichtet. Wir zeichnen diese Beziehung wie die Modellierung durch Bäume/Wälder, um so Hierarchiebeziehungen und Abläufe in der grafischen Darstellung unterscheiden zu können. Diese Bäume/Wälder entsprechen durch ihre Richtung/Hierarchie nicht mehr der formalen Definition eines Waldes in der Graphentheorie.

[65] Es sollte auf eine eingängige und konsistente Wahl der Bezeichner geachtet werden

[66] Wir bezeichnen diese Instanz als Default

Die Glass-Box- und Black-Box Sicht von Geschäftsprozessen ist modelliert durch Hierarchiebeziehungen, die zwischen zwei Geschäftsprozessen definiert sind, die schon definierte Geschäftsprozesshierarchie. Dadurch entstehen in unserem Modell Wälder, die alle Geschäftsprozesse beschreiben.

- Hierarchiebeziehungen
  Zu den einzelnen Geschäftsprozessen sind Verfeinerungsbeziehungen und Abstraktionsbeziehungen im Modell hinterlegt, die die Geschäftsprozesse in Hierarchien einteilen. Atomare Geschäftsprozesse haben keine Elemente, die sie weiter verfeinern. Geschäftsprozesse auf höchster Ebene haben keine weiteren Geschäftsprozesse, die sie abstrahieren. Die Verfeinerungsbeziehung ist gerichtet zwischen dem übergeordneten Geschäftsprozess zum untergeordneten Prozess. Die Hierarchiebeziehungen zwischen den Geschäftsprozessen ergeben in unserem Modell einen Wald. Wir lassen damit verschiedene Bäume von Geschäftsprozessen zu, um größere Modelle erstellen zu können, die die gesamte IT-Landschaft und alle Geschäftsprozesse beinhalten können.

Abhängigkeiten und Abläufe zwischen den Geschäftsprozessen sind modelliert durch Geschäftsprozessabläufe.

- Geschäftsprozessabläufe
  Beziehungen zwischen den Geschäftsprozessen modellieren die Abläufe, in denen Prozesse nacheinander ausgeführt werden.

Zwischen den Teilmodellen existieren die Beziehungen der Ebenen. Geschäftsprozesse werden mit zwei weiteren Ebenen verknüpft:

- Abbildung von Geschäftsprozessen auf Features des Funktionsmodells

  Hierbei werden die beiden Objekte der jeweiligen Ebene hinterlegt, ein Geschäftsprozess und ein Feature. Bei dieser Abbildung werden nur die jeweiligen Instanzen miteinander verknüpft[67]. Wird die Default Instanz verknüpft, gilt die Verknüpfung für alle Instanzen. Wie bereits beschrieben erfolgt dies bei der Erstellung eines Modells zwischen dem Geschäftsprozessverantwortlichem und dem Softwareentwickler.

- Abbildung von Geschäftsprozessen auf Softwarekomponenten/Anwendungen

  Hierbei werden die beiden Objekte der jeweiligen Ebene hinterlegt, ein Geschäftsprozess und eine Softwarekomponente oder eine Applikation. Auch bei dieser Abbildung werden nur die jeweiligen Instanzen miteinander verknüpft. Wird die Default Instanz verknüpft, gilt die Verknüpfung für alle Instanzen.

5.3.7 Abbildungen zwischen den Ebenen

Geschäftsprozesse können mit Features auf der Funktionsebene und mit Softwarekomponenten/Anwendungen verbunden werden. Die Entscheidung über die Abbildung der Geschäftsprozesse auf Softwarekomponenten oder Anwendungen[68] erfolgt entweder im Rahmen des Softwareentwurfs bei Individualsoftware, oder bei der Evaluierung und dem Customizing von Kaufsoftware. Durch die Abbildung der Geschäftsprozesse auf das Funktionsmodell wird die Zuordnung zwischen Geschäftsprozessen und Softwarekomponenten vereinfacht, wenn auch die Softwarekomponenten den entsprechenden Features zugeordnet werden.

Darüber hinaus kann das Modell benutzt werden, um Anforderungen an die einzelnen Softwarekomponenten zu modellieren und so in den Entwurfsprozess einzubringen.

Eine Abbildung zwischen den Teilmodellen ist auf allen Hierarchieebenen der Geschäftsprozesse möglich. Wir definieren jedoch zur Vereinfachung und eindeutigen Abbildung der Bezie-

[67] Wir erlauben dies, weil wir auch in den Features Instanzen zugelassen haben, raten aber, wie schon beschrieben, von einer intensiven Nutzung ab.

[68] Die Abbildung kann auf Softwarekomponenten oder Anwendungen erfolgen. Zur Unterscheidung zwischen Anwendung und Softwarekomponente siehe Kapitel 5.4.

hungen, dass untergeordnete Geschäftsprozesse eines bereits zugeordneten Geschäftsprozesses nicht mehr auf Softwarekomponenten/Features abgebildet werden können, es sei denn diese sind Verfeinerungen der Softwarekomponente/Features des übergeordneten Geschäftsprozesses und werden auch einer untergeordneten Softwarekomponente/Feature zugeordnet.

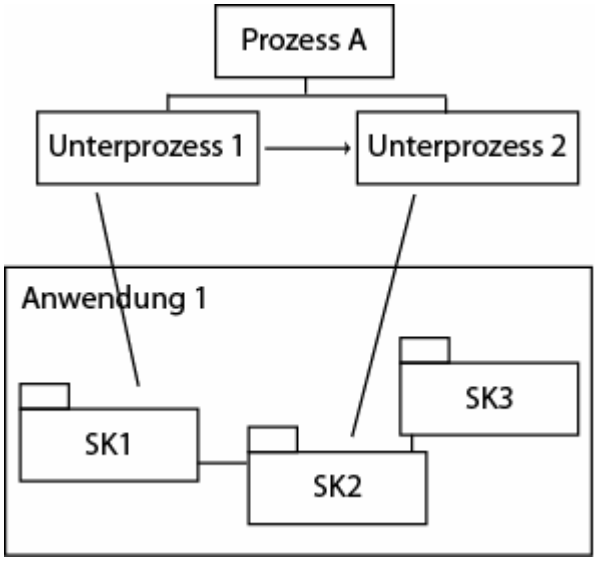

Abbildung 54 – Abbildung Geschäftsprozess auf Softwarekomponenten

Die Abbildung 54 stellt eine korrekte Modellierung dar. Ein Geschäftsprozess wird in zwei Unterprozesse verfeinert, die anschließend auf zwei Softwarekomponenten abgebildet werden. Im Gegensatz dazu verdeutlicht die Abbildung 55 eine fehlerhafte Modellierung. Hierbei wird eine Realisierungsbeziehung zwischen dem Unterprozess 1 und der Anwendung 1 modelliert, obwohl der übergeordnete Prozess durch die Teilkomponente SK2 der Anwendung 1 realisiert wird. Auch die Zuordnung des Unterprozesses 2 an die Teilkomponente SK2 ist deshalb nicht mehr notwendig und sinnvoll.

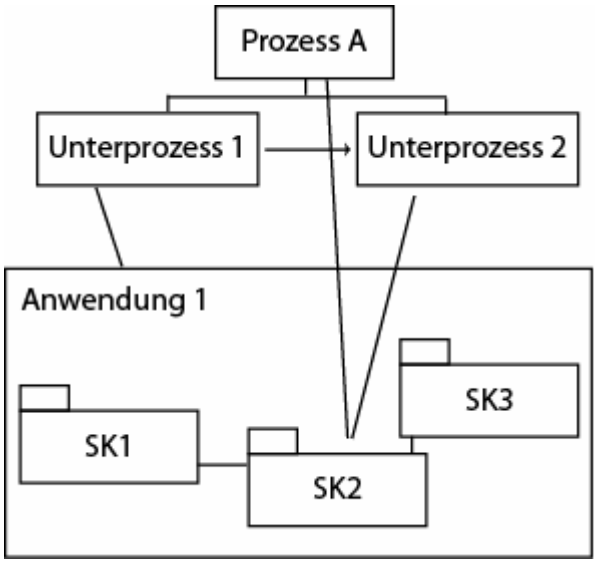

Abbildung 55 – Falsche Abbildung Geschäftsprozess auf Softwarekomponenten

Bei der Modellierung der Beziehung zwischen den Ebenen muss auf diese Zuordnung geachtet werden. Auch eine explizite Modellierung von Geschäftsprozessen, die eine Verfeinerung eines zugewiesenen Geschäftsprozesses darstellen und die selbst nicht mehr zugeordnet sind, ist meist nicht sinnvoll.

Die Granularität der Abbildung zwischen den Ebenen entscheidet, wie detailliert die Antworten des Modells sind. Eine einfache Abbildung würde den Geschäftsfall „Auftrag erfassen“ auf die Anwendung „SAP Netweaver SBS“ abbilden. In diesem Fall sind Fragen nach dem Ausfall einzelner Komponenten, bzw. Abstufungen beim Ausfall nicht möglich. Werden aber die einzelnen untergeordneten Geschäftsprozesse auf die jeweiligen Softwarekomponenten abgebildet und auch parallele Pfade hinterlegt, so lassen sich auch Fragen nach dem Einfluss des Ausfalls einzelner Softwarekomponenten beantworten.

In diesem Zusammenhang tritt ein Unterschied zwischen Kaufsoftware und Individualentwicklungen zutage. Bei der Entwicklung von Individualsoftware wird meist eine Zuordnung aller Geschäftsprozesse auf alle Softwarekomponenten entstehen, jede Softwarekomponente ist direkt oder auf höheren Ebenen einem Geschäftsprozess zugeordnet, da meist keine Funktionen entwickelt werden, die aus dem Fachkonzept der Geschäftsprozesse nicht notwendig sind. Es wird also meist eine hundertprozentige Abdeckung zwischen den Softwarekomponenten und den Geschäftsprozessen existieren.

Bei Kaufsoftware ist es möglich, dass ein größeres System eingeführt wird, um einen Geschäftsprozess zu unterstützen oder zu implementieren. Dabei ist es möglich, dass Funktionen und Softwarekomponenten noch nicht verwendet werden. Bei einem Reegineering der Geschäftsprozesse oder der Einführung neuer Geschäftsprozesse kann durch die Nutzung entsprechender Komponenten eine zusätzliche Abbildung entstehen und damit der Funktionsumfang der Kaufsoftware besser genutzt werden.

Der Grad der Nutzung der Softwarekomponenten einer Anwendung durch einen Geschäftsprozess und die Zahl der nicht auf einen Geschäftsprozess abgebildeten Softwarekomponenten kann als ein Maß für die Effektivität des Einsatzes einer Anwendung genutzt werden. Dieses Maß hängt stark vom Geschäftsprozess und dessen Wertbeitrag ab und kann nicht für verschiedene Fälle verallgemeinert werden, er kann aber bei Kaufentscheidungen und beim Vergleich verschiedener Lösungen für ein Anwendungsszenario eine maßgebliche Hilfestellung bieten.

Geschäftsprozesse ließen sich auch in den Schnittstellen der Softwarekomponenten modellieren und entsprechend in die Hierarchien eingliedern. Bei einfachen Geschäftsprozessen oder einer geringen Detaillierungstiefe kann diese Modellierung eine Vereinfachung des Modells realisieren (Siehe 6.2) Bei einer hohen Detaillierung der Geschäftsprozesse bzw. bei komplexen Beziehungen zwischen den Ebenen wie der Abbildung von Instanzen von Geschäftsprozessen auf Instanzen von Softwarekomponenten reicht diese Form der Modellierung häufig nicht mehr aus. Hier ist eine eindeutige, von den Schnittstellen losgelöste Modellierung der gegenseitigen Beziehungen notwendig. In diesem Fall sollten die Geschäftsprozesse nicht in die Schnittstellen der Softwarekomponenten modelliert werden.

5.3.8 Funktionen auf Geschäftsprozessen

In diesem Anschnitt definieren wir die Funktionen, die in unserem Geschäftsprozessmodell Elemente erstellen und Operationen auf ihnen ermöglichen. Damit werden Geschäftsprozesse in eingefügt, Beziehungen zwischen ihnen erstellt und die Objekte auf die anderen Ebenen abgebildet.

Anlegen eines Geschäftsprozesses

Die Funktion zum Anlegen eines Geschäftsprozesses definiert ein Geschäftsprozessobjekt im Modell. Dabei kann die Instanz mit angegeben werden. Wird die Instanz nicht angegeben, wird das Objekt mit der Defaultinstanz erstellt.

Bezeichner [x Instanzbezeichner] → Geschäftsprozess

createGP(„Bezeichner" [, „Instanzbezeichner"]) = GP

createGP(„Webshop") = GP1, dabei gilt:

Sei GP die Menge aller Geschäftsprozesse

$\forall\ GP_a \in\ GP: Bezeichner(GP_a) \neq Bezeichner(GP1)$

Formel 36 – Funktion zum Anlegen eines Geschäftsprozesses

Die Formel definiert einen Geschäftsprozess, den wir als Ausgang für ein kurzes Beispiel in den nächsten Funktionen verwenden werden. Hier wird zuerst eine Defaultinstanz des Geschäftsprozesses „Webshop" definiert.

Instanziierung eines Geschäftsprozesses

Instanzen eines Geschäftsprozesses ermöglichen es, denselben Prozess für unterschiedliche Kunden zu implementieren oder in unterschiedlichen Geschäftsprozessabläufen zu integrieren.

Geschäftsprozess x Instanzbezeichner → Geschäftsprozess

createInstance(Instanzbezeichner) = GP

GP1.createInstance(„TUM") = GP1[TUM]

GP1.createInstance(„LMU") = GP1[LMU]

Formel 37 – Funktionen zur Instanziierung eines Geschäftsprozesses

Die Funktion verwendet den definierten Geschäftsprozess und legt von diesem zwei Instanzen für die Kunden TUM und LMU an.

Zuweisen von Attributen an einen Geschäftsprozesses

Bei den übergreifenden Objekten des Modells haben wir bereits die Attribute eingeführt. Wir weisen nun unserem Geschäftsprozess ein Attribut zu.

Geschäftsprozess x Attributschlüssel x Attributwert → Geschäftsprozess

GP.setAttrib(Attributschlüssel, Attributwert) = GP`

Geschäftsprozess x Attributschlüssel → Attributwert

GP.getAttrib(Attributschlüssel) = Attributwert

GP1[TUM].setAttrib(„Ansprechpartner","Norbert Diernhofer")

GP1[TUM].getAttrib(„Ansprechpartner")="Norbert Diernhofer"

GP1[LMU].getAttrib(„Ansprechpartner")=""

GP1.setAttrib(„Interne Bezeichnung","GP23")

GP1[TUM].getAttrib(„Interne Bezeichnung")=GP23

GP1[LMU].setAttrib(„Interne Bezeichnung","GP23-1")

GP1[LMU].getAttrib(„Interne Bezeichnung")=GP23-1

Formel 38 – Funktionen zur Zuweisung von Attributen an einen Geschäfts-prozess

In diesem Beispiel haben wir unserem Geschäftsprozess das Attribut Ansprechpartner hinzugefügt. Darüber hinaus ist der Defaultinstanz das Attribut „Interne Bezeichnung" zugeordnet. Die Abfrage des Attributs Ansprechpartner für die zweite Instanz des Geschäftsprozesses ergibt keinen Wert. Die Abfrage des Attributs Interne Bezeichnung für

unseren Prozess ergibt den Wert, der auch für die Defaultinstanz gesetzt wurde. Die Informationen der Default-Instanz gelten so für alle Instanzen des Geschäftsprozesses.

Die Werte der Defaultinstanz können auch überschrieben werden, damit wird das Konzept der Verschattung realisiert. Am Beispiel des GP1[LMU] wird hier der Attributwert der Defaultinstanz mit einem anderen Wert belegt. Dieser Mechanismus sollte sehr sparsam eingesetzt werden, eine häufige Notwendigkeit der Überschreibung deutet drauf hin, dass Geschäftsprozesse nicht als ein Geschäftsprozess sondern durch verschiedene Geschäftsprozesse modelliert werden sollten.

Verkettung von Geschäftsprozessen zu Geschäftsprozessabläufen

Auf der Ebene von Geschäftprozessen definieren wir keine Einzelaktionen eines Geschäftsprozesses, wie dies in den EPKs von ARIS oder den Geschäftsprozessen Thurner möglich ist. Wir verbinden nur einzelne Geschäftsprozesse zu Geschäftsprozessabläufen.

Abbildung 56 – Verkettung von Geschäftsprozessen

Die Abbildung 56 verdeutlicht dies. Der Geschäftsprozess 1 wird dabei mit dem Geschäftsprozess 2 zu einem Ablauf verbunden, in dem die Geschäftsprozesse nacheinander ausgeführt werden. Die Funktion connect führt diese Operation aus.

Geschäftsprozess x Geschäftsprozess → Geschäftsprozessablauf

connectGP(Geschäftsprozess 1, Geschäftsprozess 2) → Geschäftsprozessablauf

createGP(„Geschhäftsprozess 1") = GP1

createGP(„Geschhäftsprozess 2") = GP2

connectGP(GP1,GP2)

Dabei gilt: isparent(GP1,GP2)=isparent(GP2,GP1) = false

*Geschäftsprozess → { Geschäftsprozessablauf }**

pred(GP2)=GP1;

suc(GP1)=GP2;

Geschäftsprozess x Geschäftsprozess → Bool

ispred(GP2,GP1)=true, wenn gilt: GP1 ∈ pred(GP1)

ispred(GP1,GP2)=false, wenn gilt: GP1 ∉ pred(GP1)

issuc(GP2,GP1)=false, wenn gilt: GP1 ∉ suc(GP1)

issuc(GP1,GP2)=true, wenn gilt: GP1 ∈ suc(GP1)

Formel 39 – Funktion zur Verkettung von Geschäftsprozessen

In diesem Beispiel haben wir zwei neue Geschäftsprozesse definiert und diese miteinander verbunden. Die Geschäftsprozessabläufe sind dabei für alle Instanzen eines Geschäftsprozesses definiert. Die Prozesse sind gerichtet verbunden, der Geschäftsprozess 2 wird erst nach dem Geschäftsprozess 1 ausgeführt.

Die Funktionen suc und pred geben eine Menge an Geschäftsprozessen aus, die allen Vorgängern und Nachfolgern eines Geschäftsprozesses entsprechen. Die Funktionen issuc und ispred geben einen boolschen Wert aus, wenn diese Bedingung zwischen den beiden übergebenen Geschäftsprozessen wahr ist.

Hierarchische Anordnung

Wir definieren zur besseren Strukturierung unserer Modelle eine Hierarchie auf Geschäftsprozessen. Dieses Konzept beinhaltet keine Vererbung von Eigenschaften. Wir definieren dazu die Verfeinerung und die Abstraktion von Geschäftsprozessen.

Verfeinerung

Die Verfeinerung von Geschäftsprozessen ist eine wichtige Funktion zur Modellierung, da sie es ermöglicht, verschiedene Abstraktionsebenen eines Prozesses darzustellen. Geschäftsprozesse werden dabei in Hierarchien eingeordnet und ihre einzelnen Funktionen weiter unterteilt. Abbildung 57 verdeutlicht dies am Beispiel eines Geschäftsprozesses, der in zwei Geschäftsprozesse unterteilt wird.

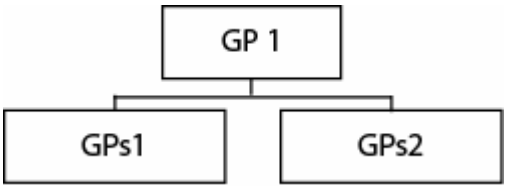

Abbildung 57 – Geschäftsprozesse und Subprozesse

Wird ein Geschäftsprozess verfeinert, so gilt diese Verfeinerung auch für alle seine Instanzen. Alle Instanzen eines Prozesses finden sich auch als Instanzen der jeweiligen Unterprozesse.

Geschäftsprozess, Bezeichner [x Instanzbezeichner] →
Geschäftsprozess x Geschäftsprozess x Hierarchiebeziehung

createsubGP(GP1, „Bestellung aufnehmen")= GPs1

Geschäftsprozess x Geschäftsprozess →
Geschäftsprozess x Geschäftsprozess x Hierarchiebeziehung

createGP(„Abrechnung durchführen","TUM") = GPs2

addsubGP(GP1, GPs2)

Dabei gilt: ispred(GP1, GPs2) = ispred(GPs2, GP1) = false

Formel 40 – Funktionen zur Verfeinerung eines Geschäftsprozesses

Durch die Funktion createsubGP wird dabei ein neuer Geschäftsprozess angelegt und dieser dem bereits existierendem Geschäftsprozess als Unterprozess zugeordnet. Die Funktion addsubGP ordnet zwei existierende Geschäftsprozesse einander in einer entsprechenden Beziehung zu. Eventuell bereits existierende Verkettungen zwischen Geschäftsprozessen bleiben von den Verfeinerungsbeziehungen unberührt und werden nicht auf die untergeordneten Prozesse übertragen.

Die Hierarchien sind nicht exklusiv. Unterprozesse eines Geschäftsprozesses können auch zu anderen Geschäftsprozessen gehören. Die damit entstehende Struktur in der Modellierung sind Wälder.

Wir definieren die Funktion children auf Hierarchien, die alle in der Geschäftsprozesshierarchie untergeordneten Geschäftsprozesse ausgibt, bzw. die Funktion parent, die alle übergeordneten Geschäftsprozesse auflöst:

Geschäftsprozess → { Geschäftsprozess } *

parent(GPs1) = GP1

children(GP1) = {GPs1, GPs2}

Formel 41 – Kindelemente einer Hierarchiebeziehung

Wir definieren Funktionen zur Überprüfung von Hierarchiebeziehungen:

Geschäftsprozess x Geschäftsprozess → Bool

ischild(GPs2, GP1) = true, wenn gilt: GPs2 ∈ children (GP1)

ischild(GP1, GPs2) = false, wenn gilt: GP1 ∉ children (GPs2)

isparent(GPs2, GP1) = false, wenn gilt: GP1 ∉ children (GPs2)

isparent(GP1, GPs2) = true, wenn gilt: GPs2 ∈ children (GP1)

isroot(GP1) = true, wenn gilt:

Sei GP die Menge aller Geschäftsprozesse

$\forall$ $GP_a \in$ *GP gilt ischild(GP1, GP_a)=false und*

E $GP_b \in$ *GP mit ischild(GP_b, GP1)=true*

isatomar(GP1) =true, wenn gilt:

Sei GP die Menge aller Geschäftsprozesse

$\forall$ $GP_a \in$ *GP gilt isparent(GP_a ,GP1)=false*

Formel 42 – Funktionen auf Hierarchiebeziehungen

Die Funktion ischild dient dabei zu prüfen, ob ein Geschäftsprozess in der Hierarchie eines anderen Geschäftsprozesses vorkommt. Die Funktion isroot gibt an, ob ein Geschäftsprozess ein Wurzelelement in einer Geschäftsprozesshierachie ist. Die Funktion isatomar gibt an, ob ein Geschäftsprozess ein Blattelement in einer Geschäftsprozesshierachie oder für sich allein stehend ist.

Abstraktion

Das Gegenstück zur Verfeinerung von Geschäftsprozessen ist die Abstraktion. Bei diesem Vorgang werden zwei Geschäftsprozesse zu einem Geschäftsprozess zusammengefasst[69]. Bei der Abstraktion könne dabei Informationen gelöscht und verloren werden. Im Gegensatz zur Verfeinerung, werden Verkettungen bei der Abstraktion auf den abstrakteren Geschäftsprozess übertragen, um die Konsistenz des Gesamtmodells sicherzustellen und „*dangling references*“ zu vermeiden. Gleiches gilt für die Abbildung auf Funktionen und Softwarekomponenten.

Geschäftsprozess [x Geschäftsprozess] x Geschäftsprozess → Geschäftsprozess

abstractGP(GP1, GP2) → GP2, falls isatomar(GP1) = true und isparent(GP2,GP1)=true;

Danach gilt:

Sei GP die Menge aller Geschäftsprozesse

$\forall$ $GP_a \in$ *GP mit issuc(GP_a, GP1) = true → connectGP(GP_a, GP2)*

$\forall$ $GP_b \in$ *GP mit ispred(GP_b, GP1) = true → connectGP(GP_b, GP2)*

pred(GP1) = suc(GP1) = {}

Formel 43 – Funktion zur Abstraktion von Geschäftsprozessen

[69] Eine Spezialform der Abstraktion ist die Realisierung einer Zoom-Funktion in einem Tool. Damit wird nicht das Modell an sich, sondern die Darstellung geändert. Die Funktionalität entspricht dabei unserer Abstraktionsfunktion, die Informationen gehen in diesem Fall aber nicht verloren, da nur die Darstellung und nicht das Modell an sich verändert wird.

Wir lassen die Abstraktion dabei nur von atomaren Geschäftsprozessen zu. Die Nachbedingung gilt in selber Weise für die Abbildung zwischen den Ebenen, die wir im nächsten Abschnitt einführen.

Abbildung zwischen den Ebenen

Die Abbildung von Geschäftsprozessen auf Softwarekomponenten und Applikationen stellt die Verbindung zwischen den beiden Ebenen dar. Gleiches gilt für die Abbildung von Geschäftsprozessen auf Features. Dabei werden einzelne Instanzen von Geschäftsprozessen auf einzelne Softwarekomponenten oder Features abgebildet. Die Verbindungen sind dabei bidirektional gerichtet. Für Features definieren wir:

Geschäftsprozess x Funktion → Geschäftsprozessabbildung

connectGPF(Geschäftsprozess, Funktion) = Geschäftsprozessabbildung

connectGPF(GPs1, „Bestellwesen")=GPAS1

connectGPF(GPs2, „Finanzwesen")=GPAS2

Formel 44 – Funktion zur Abbildung von Geschäftsprozessen auf Funktionen

Durch die Formel werden die beiden bereits eingeführten Geschäftsprozesse GPs1 und GPs2 auf die beschriebenen Features abgebildet. In diesem Fall sind dabei die betrieblichen Organisationseinheiten als Funktionen modelliert.

Die Formeln gelten analog für Softwarekomponenten und Anwendungen. Wie im Kapitel 5.4 beschrieben, bestehen Anwendungen aus Softwarekomponenten. Für die Abbildung zwischen den beiden Ebenen ist es deswegen unerheblich, ob die Abbildung des Geschäftsprozesses auf Softwarekomponenten oder eine Anwendung erfolgt.

Geschäftsprozess x Anwendung | Softwarekomponente → Geschäftsprozessabbildung

connectGPSA(Geschäftsprozess, Anwendung | Softwarekomponente) = Geschäftsprozessabbildung

connectGPSA(GPs1, Anwendung1)=GPF2

Formel 45 – Funktion zur Abbildung von Geschäftsprozessen auf Funktionen

Die Konsistenzsicherung muss dabei sicherstellen, dass keine übergeordneten Prozesse auf Softwarekomponenten oder Features abgebildet werden, die hierarchisch unterhalb den zugewiesenen Softwarekomponente oder Features angeordnet sind.

5.4 Softwaremodell

Die Softwarekomponenten und Anwendungen, die in der IT-Landschaft ausgeführt werden, werden in unserem Softwaremodell beschrieben. Es beschreibt die Softwarekomponenten und ihre gegenseitigen Interaktionen und Abhängigkeiten. Die Ebene stellt die Implementierung der Geschäftsprozesse dar. Dabei sind die einzelnen Geschäftsprozesse mit den jeweiligen Softwarekomponenten verbunden. Die Implementierung erfolgt durch den Entwickler, der die Softwarearchitektur entwirft und umsetzt, bzw. Kaufsoftware adaptiert.

Im Vergleich zu den Modellen der Softwareentwicklung entspricht dieses Teilmodell einem Architekturschema des Softwaresystems. Relevant sind die Komponenten der Architektur, die nicht auf mehrere IT-Systeme verteilt werden können. Diese Komponenten sind die kleinsten Einheiten, die in unserem Softwaremodell als Softwarekomponenten beschrieben werden.

Die Softwarearchitektur unterscheidet zwischen starker und schwacher Kopplung. Üblicherweise werden Komponenten oder Dienste, die schwach miteinander gekoppelt werden sollen, über die Implementierung des Facade Pattern[GOF95] realisiert. Damit wird eine Trennung zwischen den verschiedenen Diensten vereinfacht, indem größere Module aufgebaut werden, die besser voneinander abgegrenzt werden können. Die Schnittstellen werden über die Facade für andere Softwarekomponenten angeboten. Zur Erbringung der Dienste der Facade können mehrere Komponenten implementiert werden, die hinter dieser gelagert, die entsprechenden Funktionen realisieren. Die interne Kommunikation dieser Dienste findet meist eng gekoppelt statt. Sie wird nicht nach außen verfügbar gemacht, deshalb können entsprechende Komponenten auch nicht mit anderen verbunden werden und sind für die Softwareebene nicht relevant. In diesem Fall würden alle Komponenten, die zur Bereitstellung der Dienste der Facade notwendig sind und die keine externen Dienste im Sinne des Facade Pattern darstellen, zu einer Softwarekomponente zusammengefasst. Die Abbildung 58 verdeutlicht dies an einem Beispiel. Die drei Softwarekomponenten SK1, SK2 und SK3 stellen die Dienste der Anwendung dar und realisieren die verschiedenen Funktionen. Dienste der Softwarekomponenten SK1 und SK3 werden über die Facade auch externen Diensten angeboten. Damit ist es möglich, Zugriffe auf die internen Softwarekomponenten nur über eine definierte Softwarekomponente zuzulassen. Diese kann dann zum Beispiel sicherstellen, dass nur berechtigte Benutzer Zugriff auf die Dienste beinhalten. Dies muss dann bei den Softwarekomponente SK1 und SK3 nicht mehr extra geprüft werden. In diesem Szenario würden die Softwarekomponenten SK1, SK2, SK3 und die Facade zu einer Softwarekomponente zusammengefasst und als eine Softwarekomponente modelliert. Die interne Struktur wird in dieser Softwarekomponente versteckt, da sie für andere Komponenten nicht relevant ist.

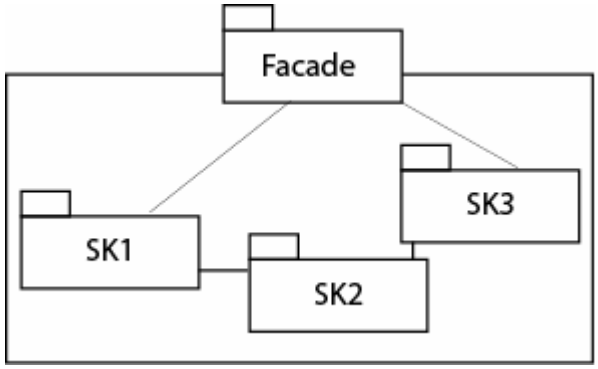

Abbildung 58 – Facade Pattern

Um die Software auszuführen, muss sie auf der IT-Infrastruktur installiert werden. Dabei werden Entscheidungen des Entwicklers konkretisiert, indem Instanzen der jeweiligen Komponenten ausgeführt werden. Möglicherweise werden bei der Installation und der Instanziierung der Software nochmals Verfeinerungen an den Schnittstellen der Komponenten durchgeführt (Siehe auch Formel 8).

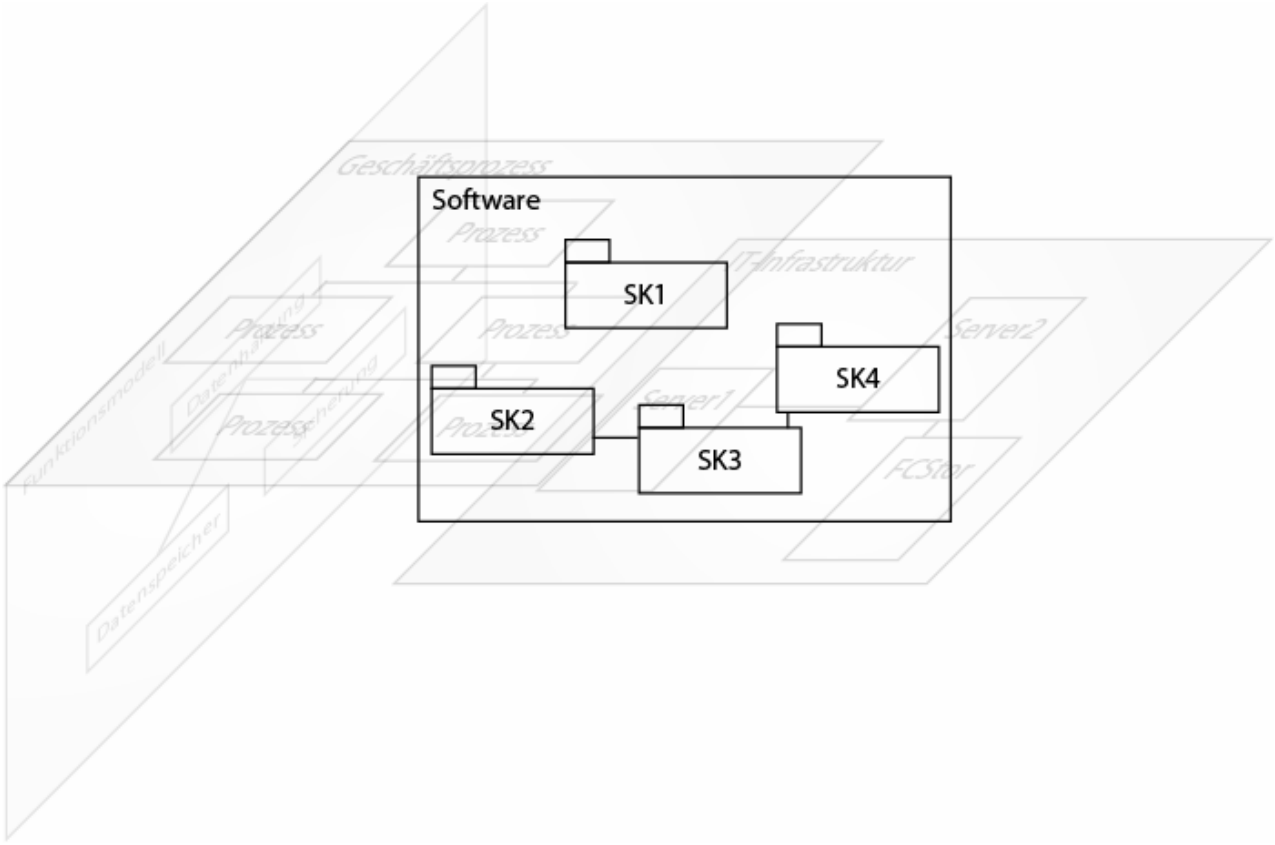

Abbildung 59 – Die Ebene der Software

Unsere Softwaremodellebene enthält verschiedene Bestandteile, die die Softwarelandschaft beschreiben. Dabei unterscheiden wir Softwarekomponenten, die atomare Komponente beschreiben, die auf einem IT-System ausgeführt werden und Anwendungen, die aus verschiedenen Softwarekomponenten bestehen.

5.4.1 Anwendung

Wir definieren eine Anwendung als:

> *Eine Anwendung ist ein Softwaresystem, dass aus einer oder mehreren Softwarekomponenten besteht und ein Systemverhalten implementiert.*

Definition 32 – Anwendung

Die Syntax ist gegeben durch:

> *Anwendung = (Bezeichner) { [Instanz] } **

Formel 46 – Syntax einer Anwendung

Die Softwarekomponenten können dabei in einer Baumstruktur dargestellt und ihre Abhängigkeiten verdeutlicht werden. Eine Anwendung dient in unserem Modell vor allem der besseren Strukturierung des Softwaremodells. Abbildung 60 stellt eine Anwendung bestehend aus 3 Softwarekomponenten dar. Die grafische Darstellung der einzelnen Softwarekomponenten orientiert sich dabei an der Klassendarstellung von UML Diagrammen. Im Gegensatz zu UML Diagrammen modellieren wir aber grafisch keine Attribute von Softwarekomponenten.

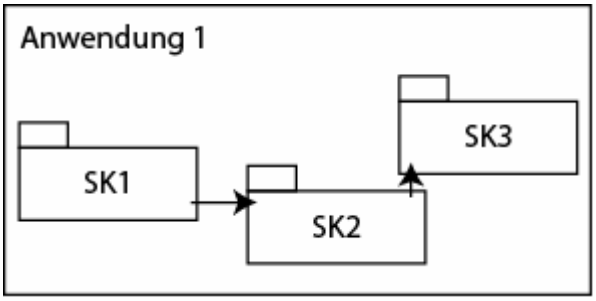

Abbildung 60 – Anwendung bestehend aus 3 Softwarekomponenten

Die Anwendung ist damit die Abgrenzung eines Softwaresystems zu anderen Softwaresystemen. Im Sinne der Arbeit beschreibt eine Anwendung Kaufsoftware oder das Ergebnis eines Projektes mit klar definierten Grenzen. Sie dient vor allem der Abgrenzung zu anderen Anwendungen und ist nur eine Hilfskonstruktion, um Softwarekomponenten einordnen zu können. Im Bezug auf die Softwarekomponenten ist eine Anwendung nur ein Container, der keine Eigenschaften, funktionalen Einschränkungen oder Grenzen der Softwarekomponenten definiert.

Ein Entwickler wird meist die Beschreibung all seiner Komponenten aber als eine Anwendung weitergeben, daher kann die Modellierung als Anwendung für den Administrator durchaus hilfreich sein, um einfach Gruppen von Softwarekomponenten voneinander abgrenzen zu können. Eine Anwendung kann mit ihren Softwarekomponenten einen Geschäftsprozess implementieren.

5.4.2 Softwarekomponente

Unsere Vorstellung einer Softwarekomponente entspricht der geläufigen Vorstellung eines Programms, das auf einen Rechner kopiert und dort ausgeführt wird.

Eine Softwarekomponente ist definiert durch:

> *Eine Softwarekomponente ist die Realisierung eines Systemverhaltens in einer maschinenlesbar- und maschinenverarbeitbaren Form. Eine Softwarekomponente kann aus mehreren untergeordneten Softwarekomponenten bestehen. Ist eine Softwarekomponenten nicht weiter unterteilbar, so bezeichnen wir diese Softwarekomponente als atomar.*

Definition 33 – Softwarekomponente

Die Syntax ist gegeben durch:

*Softwarekomponente = (Bezeichner) { [Instanz] } **

Formel 47 – Syntax einer Softwarekomponente

Softwarekomponenten können für sich alleine modelliert werden, die Modellierung innerhalb einer Anwendung zur besseren Strukturierung ist jedoch meist vorzuziehen. In der Baumdarstellung einer Anwendung finden sich atomare Softwarekomponenten als Blätter des Baums.

Abbildung 60 verdeutlicht die grafische Darstellung. In diesem Fall sind die Softwarekomponenten nicht in einer Hierarchie eingeordnet. Die Softwarekomponenten SK1 hängt von der Softwarekomponenten SK2 ab, welche wieder von der Komponenten SK3 abhängt. Alle drei sind

atomar und können beliebig verteilt werden. Die Abhängigkeiten der Software sind anhand ihrer Schnittstellen definiert. Hierarchien auf der Ebene der Softwarekomponenten dienen vor allem der Strukturierung komplexer Anwendungssysteme.

5.4.3 Beziehungen zwischen Softwarekomponenten

Die Beziehungen und Abhängigkeiten der Softwarekomponenten sind gegeben durch die Bereitstellung und Nutzung von Schnittstellen. Diese vertikalen Abhängigkeiten beschreiben damit Abläufe auf der Ebene der Softwarekomponenten. Die Beziehungen sind dabei vom Empfänger, der die Nachricht verarbeitet hin zum Sender gerichtet, der die Information bereitstellt. Der Sender kann ohne den Empfänger ausgeführt werden, der Empfänger ist aber vom Sender abhängig.

Die Abbildung 60 beschreibt ein einfaches Beispiel. Die Abhängigkeit zwischen SK1 und SK2 bedeutet, dass die Softwarekomponenten SK1 einen Endpunkt mit der Schnittstelle

(:SK2)*

benötigt, um ausgeführt werden zu können. Auf dem IT-System, das mit diesem Endpunkt verbunden ist, muss die Softwarekomponente SK2 entsprechend eingerichtet sein. Selbiges gilt für die Softwarekomponenten SK2, die einen Endpunkt mit der Schnittstelle

(:SK3)*

benötigt, der entsprechend auf dem IT-System realisiert werden muss, mit dem der Endpunkt verbunden ist.

Diese Abhängigkeiten können jedoch durch den Entwickler entsprechend eingeschränkt werden. So könnte ein Entwickler die Schnittstelle etwa

(:IP:TCP:HTTP:SOAP:SK2)*

definieren und damit die Hierarchie genauer spezifizieren. Wenn eine unterschiedliche Instanzierung der Schnittstellen möglich ist, müsste die Schnittstelle entsprechend mit

(:SK2[Instanzeinformation1|Instanzinformation2])*

modelliert werden, die Anforderung der Softwarekomponente SK1 würde entsprechend

(:SK2)*

lauten. Ein Administrator würde diese Information dann durch die Installation entsprechend vervollständigen.

Abbildung 61 veranschaulicht die Kommunikation an einem erweiterten Beispiel. Ein Sender schickt per Multicast eine Nachricht an mehrere Empfänger. Für unser Modell ist die Beziehung dadurch modelliert, dass sowohl Empfänger1 als auch Empfänger2 die Schnittstelle von Sender1 als Anforderung modelliert haben. Die Tatsache, ob die Nachricht technisch als eine Nachricht oder zweimal übertragen wird, ist für das Modell unerheblich, da wir an den Abhängigkeiten an sich interessiert sind.

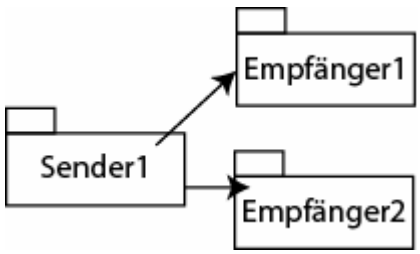

Abbildung 61 – Beziehung in Multicast Szenario

Die Anforderungen und die angebotenen Schnittstellen werden bereits bei der Modellierung der Softwarekomponenten festgelegt. Diese müssen beim späteren Deployment nicht mehr an die

konkreten Schnittstellen angepasst werden, im Sinne einer einfachen Handhabung des Modells sollte dies jedoch erfolgen[70].

Softwarekompoente | Anwendung x Schnittstelle →
Softwarekomponente | Anwendung

Empfänger1.provides(:Dienst1)*

Empfänger2.provides(:Dienst2)*

Sender1.requ(:Dienst1)*

Sender1.requ(:Dienst2)*

Formel 48 – Beziehungen zwischen Softwarekomponenten

Eine Unterscheidung, ob die Kommunikation synchron oder asynchron erfolgt ist nur dann von Bedeutung, wenn eine Middleware die Komponenten entkoppelt, da diese dann modelliert werden muss. Ist die Kommunikation in den Softwarekomponenten eingebettet, existiert für die Beziehung kein Unterschied, da wir die Abhängigkeit an sich modellieren. Die Unterschiede finden sich dann in den Funktionen, die beim jeweiligen Ausfall ausgeführt werden. Diese berücksichtigen dann eventuell vorhandene Puffer und Nebenläufigkeiten der Systeme.

5.4.4 Instanzen auf der Softwareebene

Instanzen können auf der Ebene der Software sowohl von den Softwarekomponenten selbst, als auch von den Geschäftsprozessen, die sie implementieren, benötigt werden. Ein Anbieter von Outsourcing-Dienstleistungen, der Lösungen für mehrere Kunden anbietet, würde demnach mehrere Instanzen der Implementierung für die verschiedenen Kunden ausführen. Unser Modell erzwingt jedoch nicht, dass diese Instanzen auch auf der Ebene der Software existieren oder modelliert werden müssen, falls dies in der technischen Realisierung nicht der Fall ist. So lassen sich verschiedene Instanzen eines Geschäftsprozesses auch auf eine Instanz einer Softwarekomponente abbilden.

Wie bei Geschäftsprozessen definieren wir eine Default-Instanz, die alle Instanzen einer Softwarekomponente oder einer Anwendung zusammenfasst. Existieren benannte Instanzen einer Anwendung/Softwarekomponente, dient die Default-Instanz auch hier nur noch dem Zugriff. Gleiches wie bei Geschäftsprozessen gilt für die Verschattung von Attributen.

5.4.5 Anforderungen auf der Softwareebene

Anforderungen können sich auf der Softwareebene sowohl aus fachlichen Anforderungen der Geschäftsprozesse, als auch aus technischen Anforderungen der Softwarekomponenten selbst oder der benötigten Infrastrukturdienste ergeben. Ein Portalserver[71] benötigt normalerweise eine Konfigurationsdatenbank, in der alle Einstellungen gespeichert werden. Diese Anforderung stellt eine technische Anforderung dar und wird nicht aus den Geschäftsprozessen abgeleitet, sondern ergibt sich aus dem Einsatz des Portalservers. Die Anforderung zur Bereitstellung von Inhaltsdatenbanken für die jeweiligen fachlichen Portale ergibt sich aus den Geschäftsprozessen, die der Portalserver realisiert.

Die Anforderungen werden zu den einzelnen Softwarekomponenten als Req-Attribute modelliert. In den Anforderungen werden die entsprechenden Schnittstellen, die benötigt werden, hinterlegt.

---

[70] Die Konsistenz des Modells ist auch bei der allgemeinen Modellierung gegeben, da die Anforderungen über die Schnittstellen eindeutig an eine Verbindung und einen Kommunikationspartner gebunden sind.

[71] Das Beispiel basiert auf dem Microsoft Office Sharepoint Portal Server 2003, ist aber auf andere Produkte und andere Anwendungen übertragbar.

5.4.6 Funktionen auf Softwareebene

Wir definieren Funktionen zum Umgang mit Softwarekomponenten und Anwendungen in unserem Modell.

Bezeichner [x Instanzbezeichner] → Anwendung

createAnwendung(„Bezeichner" [, „Instanzbezeichner"]) = Anwendung

createAnwendung(„Webserver")=Anw1, dabei gilt:

Sei Anw die Mange aller Anwendungen,

$\forall\ Anw_a \in Anw: Bezeichner(Anw_a) \neq Bezeichner(Anw1)$

Formel 49 – Funktion zum Anlegen einer Anwendung

Mit der Funktion createAnwendung wird eine Anwendung definiert, durch die Angabe eines Instanzbezeichners kann dabei die Instanz definiert werden. Entfällt der Instanzbezeichner, wird die Default-Instanz definiert. Die Funktion createSK gilt analog für Softwarekomponenten.

Bezeichner [x Instanzbezeichner] → Softwarekomponente

createSK(„Bezeichner" [, „Instanzbezeichner"]) = SK

createSK(„http-demon") = SK1, dabei gilt:

Sei SK die Menge aller Softwarekomponenten,

$\forall\ SK_a \in SK:\ Bezeichner(SK_a) \neq Bezeichner(SK1)$

Formel 50 – Funktion zum Anlegen einer Softwarekomponente

Wie bereits bei Geschäftsprozessen beschrieben, verwenden wir auf allen Ebenen Instanzen zur Beschreibung von Komponenten, die mehrmals in unserem Modell existieren. Die entsprechenden Funktionen auf der Ebene der Software sind gegeben durch:

Anwendung | Softwarekomponente x Instanzbezeichner →
Anwendung | Softwarekomponete

createInstance(Instanzbezeichner) = Anwendung | Softwarekomponente

Anw1.createInstance(„Firmenwebsite") = Anw1[Firmenwebsite]

SK1.createInstance(„Firmenwebsite") = SK1[Firmenwebsite]

Formel 51 – Funktionen zur Definition einer Instanz im Softwaremodell

Die Zuordnung von Attributen zu Softwarekomponenten und Anwendungen erfolgt analog wie bei den Geschäftsprozessen in Formel 38 definiert. Wir verzichten hier auf genauere Beispiele für die Funktionen. Wir weißen der Anwendung ein Requirement zu. Die Anwendung Anw1 benötigt hier eine Anwendung, die eine LDAP Schnittstelle zur Verfügung stellt.

Softwarekomponente | Anwendung x Attributschlüssel x Attributwert →
Softwarekomponente | Anwendung

Anw1.setAttrib(„req", ":LDAP")*

Anw2.setAttrib(„impl", ":LDAP")*

Formel 52 – Funktionen zur Zuweisung von Attributen an Softwarekomponenten

Die Anwendung Anw2 realisiert diese Schnittstelle. Damit ist ein möglicher Ablauf zwischen diesen beiden Anwendungen auf der Ebene der Software modelliert. Endgültig definiert wird dieser Ablauf dann beim Deployment der Anwendungen auf der IT-Infrastruktur und der damit verbundenen Konkretisierung der Schnittstellen.

Die Zuordnung von Softwarekomponenten zu Anwendungen erfolgt äquivalent zu den Hierarchien der Geschäftsprozesse. Wichtig ist hierbei zu beachten, dass eine Anwendung in einer Hierarchie nicht einer Softwarekomponente untergeordnet werden kann, Anwendungen aber in Hierarchien angeordnet werden können.

Anwendung x Anwendung →
Anwendung x Anwendung x Hierarchiebeziehung

addsubAnw(Anw1, Anw2)

Anwendung x Softwarekomponente →

Anwendung x Softwarekomponente x Hierarchiebeziehung

addsubSK(Anw1, SK1)

Softwarekomponente x Softwarekomponente →
Softwarekomponente x Softwarekomponente x Hierarchiebeziehung

addsubSK(SK1, SK2)

Formel 53 – Funktion zur hierarchischen Anwendung von Softwarekomponenten

Die Funktionen isroot, isparent und ischild definieren wir analog zu Formel 42 auf Geschäftsprozessen auf für unser Softwaremodell.

*Anwendung | Softwarekomponente → { Anwendung | Softwarekomponente }**

parent(SK1) = Anw1

children(Anw1) = SK1

ischild(SK1, Anw1) = true, wenn gilt: SK1 ∈ children (Anw1)

ischild(Anw1, SK1) = false, wenn gilt: SK1 ∉ children (Anw1)

isparent(SK1, Anw1) = false, wenn gilt: Anw1 ∉ children (Sk1)

isparent(Anw1, SK1) = true, wenn gilt: SK1 ∈ children (Anw1)

isroot(Anw1) = true, wenn gilt:

Sei Anw die Menge aller Anwendungen

$\forall$ $Anw_a \in$ *Anw gilt ischild(Anw1, Anw_a)=false und*

E $Anw_b \in$ *Anw mit ischild(Anw_b, Anw1)=true*

isatomar(SK1) =true, wenn gilt:

Sei SK die Menge aller Softwarekomponenten

$\forall$ $SK_a \in$ *SK gilt isparent(SK_a ,SK1)=false*

Formel 54 – Funktionen auf Hierarchiebeziehungen

Es wäre wünschenswert, auf der Ebene der Softwarekomponenten und Anwendungen nur eindeutige Hierarchien zuzulassen. Damit wäre jede Softwarekomponente auf einem IT-System eindeutig einer Applikation zugeordnet, bei deren Installation die Komponente auf das System gebracht wurde.

Leider ist diese Vorgehensweise gerade bei komplexen Serverapplikationen nicht möglich. Viele Anwendungen beinhalten Softwarekomponenten, die von ihnen verwendet werden aber auch von anderen Anwendungen installiert werden und häufig auch zum Betriebsystem gehören. Diese Softwarekomponenten haben also nicht nur eine vertikale Beziehung zwischen den Soft-

warekomponenten, sondern sind auch in die Hierarchie der Anwendung eingebunden. Ein Beispiel dafür sind Schnittstellen und Treiber zu Datenbanksystemen. Die Modellierung dieser zusätzlichen Komponenten ist wichtig, weil sie bei der Entscheidung, auf welchem IT-System eine Anwendung installiert werden kann, eine Rolle spielen können[72].

Die Abläufe sind auf der Ebene von Softwarekomponenten über die Reqirement-Attribute modelliert. Eine Abstraktion beeinflusst auf dieser Ebene also die Attribute der Softwarekomponente oder Applikation die die Abstraktion realisiert.

Softwarekomponente [x Softwarekomponente] x Softwarekomponente → Softwarekomponente

abstractSK(SK1, SK2) → SK2, falls isatomar(SK2) = true und isparent(SK2,SK1)=true;

Danach gilt:

$SK2.req = SK2.req \cap SK1.req$

Formel 55 – Funktion zur Abstraktion von Softwarekomponenten

5.4.7 Beispiel einer Softwarekomponente

Die Abbildung 62 zeigt einen Emailserver, bestehend aus seinen Softwarekomponenten. Dabei sind die Komponenten IMAP, Storage, POP3, Spamfilter, SMTP, eSMTP und UUCP atomar und können jeweils auf verschiedene IT-Systeme verteilt werden. Darüber hinaus sind Abhängigkeiten definiert, die die Interaktion der Softwarekomponenten beschreiben.

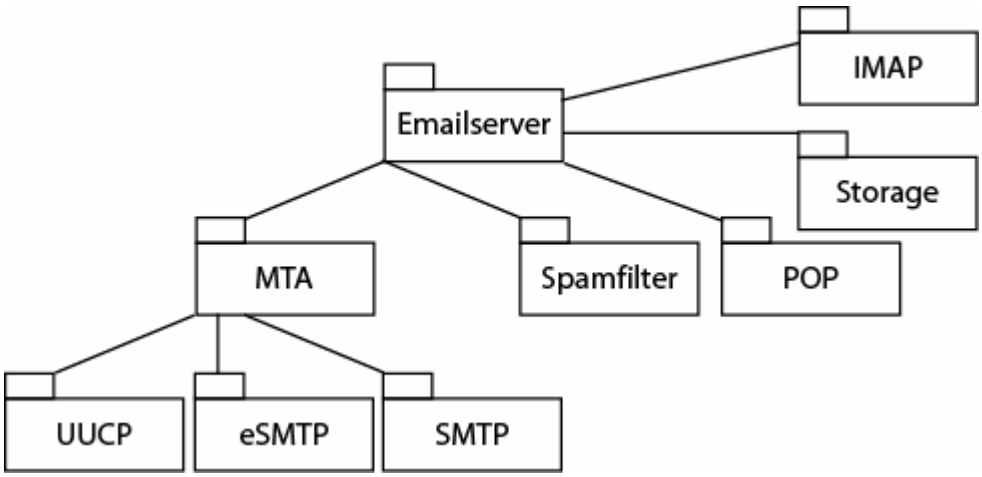

Abbildung 62 – Emailserver

[72] Gerade Sicherheitsabwägungen können hier dazu führen, eine spezielle Anwendung nicht auf einem Server zu installieren, weil sie zusätzliche Softwarekomponenten auf ein System bringen würde.

Storage.req=(„Server.Stor>1GB")

Storage.provides=(„.EMAILStorage")*

IMAP.req=(„:EMAILStorage")*

POP3.req=(„:EMAILStorage")*

eSMTP. provides =(„:EMAILMTA:eSMTP")*

SMTP. provides =(„:EMAILMTA:SMTP")*

UUCP. provides =(„:EMAILMTA:UUCP")*

Spamfilter.req=(„.EMAILMTA") and ("*.EMAILStorage")*

Bei der Installation der Anwendung auf einem IT-System werden die entsprechenden Schnittstellen dann an die jeweiligen Endpunkte gebunden[73].

5.4.8 Abbildung zwischen den Teilmodellen

Die Softwarekomponenten stehen in der „Mitte“ unseres Modells. Sie bringen die Geschäftsprozesse in eine maschinenverwertbare Form, die auf der IT-Infrastuktur ausgeführt werden kann. Sie sind verbunden mit den Geschäftsprozessen, die sie realisieren, den Features, denen sie zugeordnet sind und den IT-Systemen, auf denen sie ausgeführt werden.

Um das Geschäftsprozessmodell und das Softwaremodell miteinander zu verbinden, werden die einzelnen Geschäftsprozesse auf die Softwarekomponenten oder Anwendungssysteme abgebildet. Die Funktionen dazu haben wir bereits im Kapitel 5.3.8 beschrieben.

Um Softwarekomponenten und Anwendungen mit Features zu verbinden, werden die folgenden Funktionen verwendet:

Anwendung | Softwarekomponente x Feature → Featurebindung

connectSF(Softwarekomponente, Feature) = Featurebindung

Formel 56 – Funktion zur Abbildung von Softwarekomponenten/Anwendungen auf Features

Die Abbildung von Softwarekomponenten auf IT-Systeme, also das Deployment von Software, definieren wir im Abschnitt IT-Infrastruktur 5.6.

[73] Auf die Modellierung von Anforderungen an die IT-Infrastruktur wurde zur Vereinfachung des Beispiels verzichtet.

5.5 Infrastrukturmodell

Die Infrastruktur mit den IT-Systemen und den physikalischen Netzwerkkomponenten wird beschrieben im Infrastrukturmodell. Es beschreibt die physikalische Hardware, auf der die Softwarekomponenten ausgeführt werden. Dazu gehören auf dieser Ebene Serversysteme, Netzwerkverbindungen sowie dazu notwendige Komponenten, aber auch virtualisierte Server oder Speichersubsysteme.

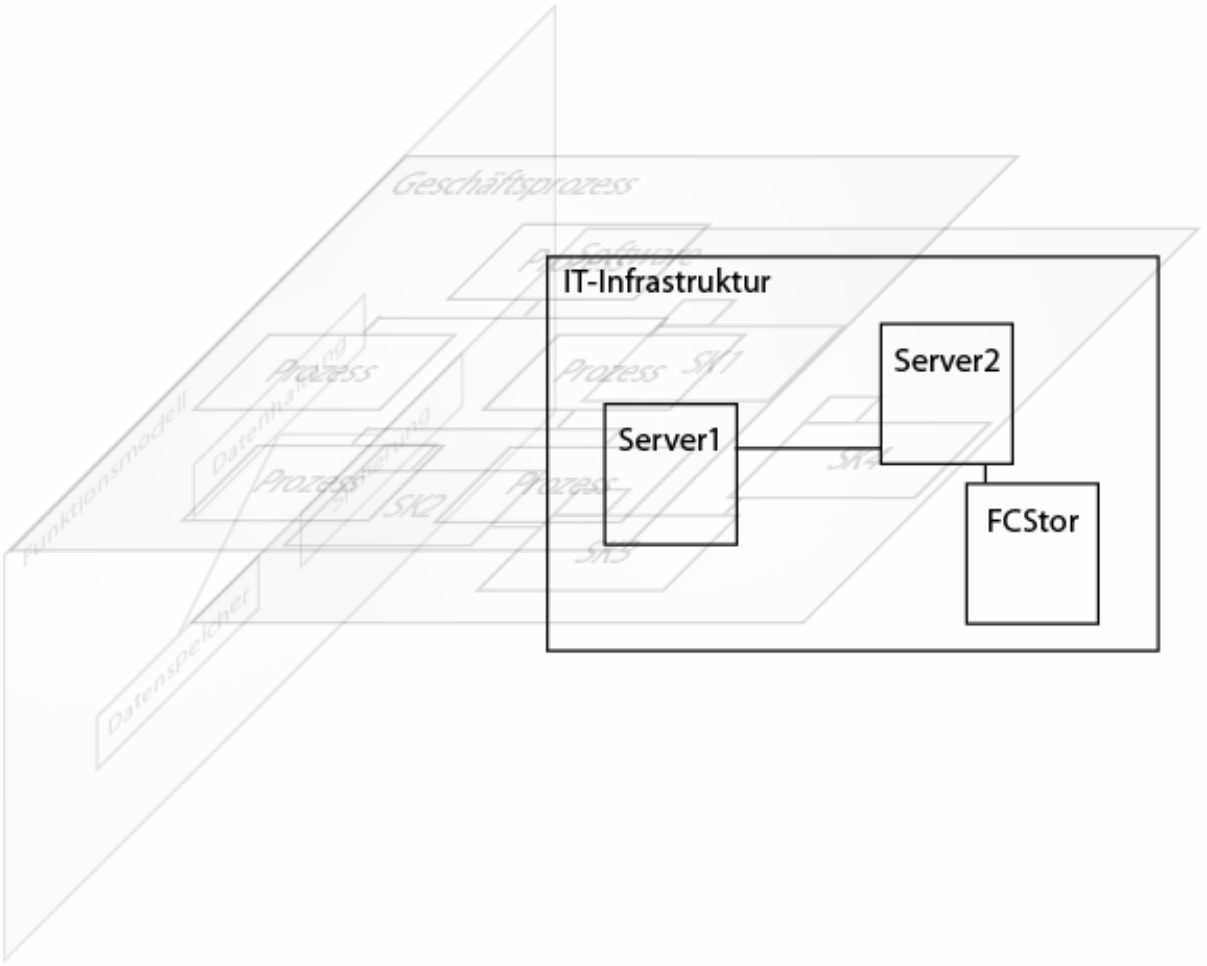

Abbildung 63 – Die Ebene der IT-Infrastruktur

Das Infrastrukturmodell beschreibt die Umgebung, in der Anwendungen bzw. ihre Softwarekomponenten ausgeführt werden können. Zur Infrastruktur zählen:

- IT-Systeme
- Kommunikationsinfrastruktur
  - Aktive Netzwerkkomponenten
  - Physikalische Kanäle

Wir werden in diesem Abschnitt die einzelnen Elemente darstellen.

5.5.1 IT-Systeme

IT-Systeme beschreiben die Hardware, also Serversysteme sowie deren Bestandteile innerhalb unseres Modells.

Wir definieren ein IT-System:

> *Ein IT-System ist eine physikalische oder logische Komponente, die Bestandteil einer IT-Infrastruktur ist, auf der Softwarekomponenten ausgeführt werden können. Ein IT-System besitzt Endpunkte, über die es durch Kanäle mit anderen Systemen verbunden werden kann. IT-Systeme können andere IT-Systeme in einer Hierarchie beinhalten.*

Definition 34 – IT-System

Die Syntax ist gegeben durch:

*IT-System = (Bezeichner) { [Instanz] } **

Formel 57 – Syntax eines IT-Systems

Ein IT-System beschreibt eine aktive Komponente der IT-Infrastruktur, die als Container für Softwarekomponenten dient und diese ausführen kann. Ein Beispiel sind Server, auf denen Basissoftwarekomponenten wie Betriebsysteme oder Anwendungsframeworks (Java-Runtime, J2EE Container, .NET Framework…) ausgeführt werden. Sie stellen damit eine Laufzeitumgebung für Softwarekomponenten zur Verfügung. Sie sind untereinander über die Kommunikationsinfrastruktur verbunden. IT-Systeme können ineinander verschachtelt werden. Damit wird das Konzept der Partitionierbarkeit von Serversystemen bzw. virtuelle Server wie sie mit Hypervisorsystemen[74] realisiert werden, modelliert. Aber auch Hardwarekomponenten eines Servers wie Plattensubsysteme lassen sich so modellieren.

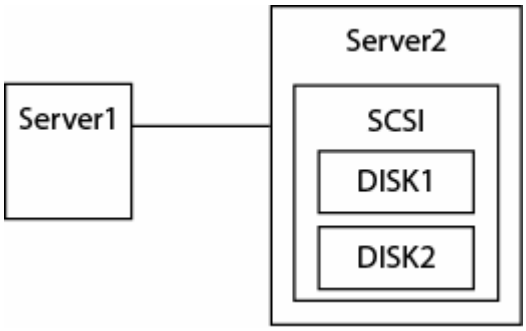

Abbildung 64 – 2 Server mit Heartbeat und SCSI Subsystem[75]

Die Abbildung 64 verdeutlicht eine entsprechende Modellierung. Zwei Serversysteme sind mit einem Netzwerkkabel verbunden. Server 2 hat ein internes SCSI Plattensubsystem, das zwei Festplatten enthält.

5.5.2 Grafische Darstellung

Grafisch modellieren wir IT-Systeme als Kasten. Entsprechend finden sich die Hierarchien als Kästen in Kästen. Wichtige Attribute von IT-Systemen lassen sich innerhalb des Kastens als Text darstellen wie in Abbildung 70 dargestellt. Diese grafische Darstellung entspricht auch den physikalischen Gegebenheiten eines Servers. Der Kanal zwischen den beiden Systemen wird durch eine ungerichtete Linie symbolisiert, da die Kommunikationsinfrastruktur bidirektional ist.

[74] Beispiele sind die Produkte von VMWare, Microsoft Virtual PC/Server, Linux XEN.

[75] Der Abbildung fehlen zur Vereinfachung die Schnittstellen der beiden Systeme.

5.5.3 Instanzen auf der Infrastrukturebene

Wir handhaben Instanzen in unserem Infrastrukturmodell im Gegensatz zu den anderen Modellebenen anders. Wir lassen die aus den anderen Ebenen bekannten Instanzen zwar zu, raten aber von deren Verwendung ab. Das Konzept der Instanzen wäre auf der Ebene der IT-Infrastruktur möglich, um gleiche Server durch Instanzen zu beschreiben. So wäre es möglich, eine Bladefarm durch verschiedene Instanzen eines Blades zu modellieren. Wird die Bladefarm als Cluster verwendet, muss diese Konfiguration jedoch extra als entsprechender Endpunkt modelliert werden. Der Vorteil der Instanzen ist also auch hier nicht gegeben.

Wir verwenden die Instanzen für IT-Systeme, um die Endpunkte eines IT-Systems zu beschreiben. Am Beispiel aus Abbildung 64 ergibt sich hier:

Server1[FC]

In der langen Schreibweise ergäbe sich hier

Server1[default][FC]

Formel 58 – Instanzen auf IT-Systemen

Wir verwenden im Weiteren die Instanzen auf IT-Systemen zur Modellierung von Endpunkten eines IT-Systems und empfehlen diese Vorgehensweise für all unsere Modelle. Als Bezeichner der Instanz des IT-Systems verwenden wir dabei den Schnittstellenbezeichner auf der niedrigsten Ebene eines Endpunktes[76].

5.5.4 Kommunikationsinfrastruktur

Die Kommunikationsinfrastruktur ist die physikalische Einheit, auf der die abstrakt modellierten Nachrichten des Geschäftsprozesses übertragen werden. Zur Infrastruktur gehören Kanäle, wie Kabel und Funkverbindungen, sowie aktive Netzkomponenten wie Switches, Router und Firewalls. Kanäle haben dabei keinen Einfluss auf die übertragenen Daten. Sie stellen nur ein Medium zur Datenübertragung zwischen verschiedenen Endpunkten zur Verfügung und realisieren damit die unterste Schicht der Übertragung in den jeweiligen Kanal- und Schnittstellenhierarchien.

Aktive Netzkomponenten der Kommunikationsinfrastruktur unterscheiden sich von IT-Systemen dadurch, dass sie keine Anwendungen ausführen, sondern eine Verbindung zu einem IT-System herstellen, an welches die Schnittstelle gebunden ist. Dazu speichern die Netzkomponenten eine entsprechende Zuordnung. Eine Netzwerkkomponente hat nur einen Endpunkt, an den alle Schnittstellenzuordnungen gebunden werden[77]. Dabei können mehrere Netzkomponenten aneinander gebunden sein. Sie stellen zusammen den Kanal zwischen einem IT-System her, das eine Schnittstelle benötigt und dem IT-System, das die Schnittstelle anbietet. Wie IT-Systeme können auch Netzwerkkomponenten Firewallregeln implementieren.

Wir verdeutlichen dies am Beispiel eines Switches. Ein Layer 2 Switch realisiert die Verbindung zwischen den IT-Systemen bis zum Layer 2. Er hat damit eine Schnittstellenhierarchie, die er realisiert. Entsprechend werden zwischen dem Switch und allen an ihm angeschlossenen IT-Systemen Kanäle modelliert. Verbindungen auf höheren Elementen der Schnittstellenhierarchie sind für die angeschlossenen Systeme direkt möglich, stützen sich aber auf den zugrunde liegenden physikalischen Kanälen ab. Auf übergeordneten Ebenen werden damit alle Schnittstellenelemente einer Schnittstellenhierachie an den Endpunkt des Switches gebunden. Dabei entspricht die Liste der Elemente der Schnittstellen auf der jeweiligen Hierarchieebene der Identität aller Schnittstellen aller mit dem Switch verbundenen Endpunkte. Damit ist ein Durchschalten der entsprechenden Kanäle zu den jeweiligen IT-Systemen je nach angeforderter Schnitt-

[76] Dies entspricht bei einem Ethernet Netzwerk der MAC Adresse der Netzwerkkarte, bei anderen Topologien wie FC der WWNID.

[77] Definiert die Netzwerkkomponenten verschiedene Netzbereiche, wie VLAN's an einem Switch, so wird pro entsprechendem Netzbereich ein Endpunkt modelliert.

stelle möglich. Darüber hinaus ist der Raum aller erreichbaren Schnittstellen angegeben. Werden zwei Switches miteinander verbunden, ergibt die Schnittmenge ihrer beiden Schnittstellen den neuen Schnittstellenraum beider Switches. Dies setzt voraus, dass die Verbindung der beiden Switches sinnvoll war, dass also die Menge der Schnittstellen der beiden Switches disjunkt war[78].

Switche erlauben mit dem Konzept der VLAN's die Partitionierung von Netzen auf einer logischen, von der Hardware losgelösten Ebene. Dieses Konzept löst sich durch den Einsatz von Instanzen des Switches im Modell abbilden und entspricht damit den Instanzen von IT-Systemen, die an die jeweiligen Endpunkte gebunden sind. Jede Instanz des Switches hat ihre eigene Menge an Schnittstellen und angebundenen Endpunkten. Auf eine Modellierung in Form eines Hierarchiekonzeptes wird zur Vereinfachung verzichtet. Kommen Switche zum Einsatz, die auch höhere Ebenen beeinflussen, kann die Modellierung als IT-Systeme sinnvoller sein.

Im Gegensatz zu Switches spannen Router meist einen deutlich größeren Adressraum auf. Vom Prinzip her entspricht die Modellierung eines Routers der Modellierung eines Switches. Ein Router wird jedoch meist Firewallregeln implementieren. Darüber hinaus ist es sinnvoll, die Schnittstellentabelle eines Routers manuell zu beschreiben und nicht als die Identität der Schnittstellen zu modellieren. Formel 59 zeigt hier die mögliche Schnittstellendefinition eines Routers, der als Standardgateway ins Internet fungiert, dabei nur http und https Zugriffe zulässt und auf Applikationsebene das Protokoll http prüft (Schnittstellenhierarchie 1). Darüber hinaus stellt er eine Anbindung an einen externen Dienstleister per DCOM her (Schnittstellenhierarchie 2).

(Ethernet:IP:TCP[80|443]:http|https

Ethernet:IP[131.159.0.0]:TCP[135])

Formel 59 – Schnittstellendefinition eines Routers

Auf der Ebene der Software sind Kanäle implizit durch die Bereitstellung und die Verwendung einer Schnittstelle durch Softwarekomponenten gegeben. Auf der Ebene der Infrastruktur werden die Kanäle als Verbindungen zwischen Endpunkten von IT-Systemen und eventuell Netzwerkkomponenten realisiert. Möglicherweise ist ein Kanal aber mit Anforderungen modelliert. Diese Anforderungen müssen auf der Infrastrukturebene dann vom Kanal als Ganzes modelliert werden. Hat der Entwickler zum Beispiel für eine Videoanwendung eine Mindestbandbreite des Netzwerkes definiert, so müssen alle Kanalabschnitte diese Anforderung erfüllen. Gleiches gilt für Eigenschaften wie Verzögerungsraten oder Verfügbarkeit. Diese Informationen werden dann auf den logischen Kanal zwischen dem Server und dem Client modelliert.

Aktive Netzwerkkomponenten behandeln Instanzen wie IT-Systeme. Passive Netzwerkkomponenten werden durch Kanäle modelliert, die keine Instanzen besitzen. Deshalb verwenden wir den Begriff Netzwerkkomponenten in den folgenden Abschnitten Synonym für aktive Netzwerkkomponenten und sprechen bei passiven Netzwerkkomponenten von Kanälen.

5.5.5 Modellierung von Ausfällen

Neben der Modellierung der Abhängigkeiten der verschiedenen Ebenen, erlaubt unser Modell auch die Modellierung von Reaktionen auf Ausfälle. Dazu werden Funktionen beschrieben, die nach dem Ausfall einer Komponente das weitere Verhalten beschreiben. Wir verwenden als Beschreibungssprache für diese Funktionen eine an Java/C# angelehnte Syntax.

Die Funktion Fail eines IT-Systems gibt die Auswirkungen auf den Ausfall einer Komponente an. Dabei ist es nicht die Aufgabe unseres Modells, entsprechende Warnmeldungen oder Informationen für ein Monitoringsystem zu generieren oder den Ausfall erkennen zu können. Das Modell soll die weiteren Auswirkungen des Ausfalls beschreiben. Damit wird ein Impact- und

[78] Diese Einschränkung kann bei erweiterten Netzkonzepten wie Switch Fault Tolerance durch eine Spanning Tree Komponente wiederhergestellt werden. In diesem Fall ist eine vorübergehende Verletzung der Anforderung möglich.

Change-Management ermöglicht, indem Änderungen und Ereignisse simuliert werden können. Dazu müssen die entsprechenden Fail-Funktionen der einzelnen Komponenten definiert werden.

Wir geben exemplarisch die Definition der Fail-Funktion eines Plattensubsystems an. Das Plattensystem besteht aus mehreren Festplatten, die gemeinsam einen Datenbereich zur Verfügung stellen. Dabei ist eine Festplatte als Hot-Spare definiert. Die anderen Festplatten befinden sich in einem RAID5 Verband. Im dieser Konfiguration werden n-1 Festplatten für die Nutzdaten verwendet und auf einer Festplatte Prüfsummen gespeichert. Fällt eine der Festplatten aus, so lassen sich die Inhalte aus den Prüfsummen errechnen, das Gesamtsystem befindet sich im sogenannten Degraded Modus. Durch das Hinzufügen einer neuen Festplatte, wird wieder der Ursprungszustand erreicht. Fällt im Degraded Modus erneut eine Festplatte aus, gehen Daten verloren. Diese Auswirkungen werden in der fail Funktion des Festplattensubsystem und der einzelnen Festplatten modelliert. Jede Festplatte ist dabei als ein IT-System mit folgender fail-Funktion modelliert:

```
fail() {
    super.fail(this.ID);
}
```

Formel 60 – Fail Funktion RAID Festplatte

Diese Funktion ruft dabei die Fehlerfunktion des gesamten Plattensubsystems auf und behandelt lokal keine Fehler. Das Gesamtsystem ist ebenfalls ein IT-System unseres Modells, das alle Festplatten als entsprechende IT-Systeme beinhaltet. Die Fail-Funktion des Gesamtsystems lautet:

```
faildisk(ID) {
if this.hotsparecount > 0 then {
    addtoRAID(selectfromhotspare());
    rebuildRAID();
    this.hotsparecount=this.hotsparecount – 1;
else {
    if (this.degraded=false) {
            this.degraded=true;
    }
    else {
            super.fail(this.ID);
    }
this.requires(this,all);
}
```

Formel 61 – Fail Funktion des Speichersubsystems

Die Fail-Funktion des Speichersubsystems ist komplexer und enthält bereits Reparaturmaßnahmen. Fällt die erste Festplatte in dieser Konfiguration aus, so wird die Hot-Spare Platte aktiviert und der RAID-Verbund reaktiviert. Steht keine Hot-Spare Platte mehr zur Verfügung, so wird das System in den Zustand degraded gesetzt. Fällt in diesem Zustand erneut eine Festplatte aus, so wird die Fail-Funktion des Containers aufgerufen. Auf der nächsten Ebene ließe sich nun eine Spiegelung zwischen Speichersubsystemen modellieren, die den Ausfall eines Speichersubsystems abfangen kann.

In allen Fällen wird am Ende die Requires Funktion aufgerufen, um eventuell modellierte Anforderungen an das IT-System zu prüfen. Eventuell modellierte Anforderungen an Ausfalltoleranz können hier dazu führen, dass eine Softwarekomponente auf dem System nicht mehr ausgeführt werden kann, obwohl das System technisch noch fehlerfrei und ohne Einschränkung arbeitet.

Diese Funktion ist ein Beispiel. Der hier beschriebene Algorithmus kann auf viele andere Komponenten erweitert werden. Durch das theoretische Durchlaufen der entsprechenden Fail-Funktionen lassen sich sehr einfach die Auswirkungen des Ausfalls verschiedener Komponenten eines Modells analysieren.

Die Fail-Funktionen unseres Modells sind hierarchisch gegliedert und finden sich auf allen Ebenen wieder. Dabei definiert die Funktion, ob die nächste Hierarchieebene aufgerufen wird (durch die Funktion super.fail), oder aber ob die Funktion selbst den Fehler so behandeln konnte, dass er keine weiteren Auswirkungen auf höhere Ebenen hat. Dabei können auch Änderungen im Laufzeitverhalten wichtig sein. Das nächste Kapitel zeigt entsprechende Beispiele. Kann eine Fail-Funktion einen Fehler zwar auflösen, führt dies aber zu einer Veränderung des Laufzeitverhaltens oder der Eigenschaften eines IT-Systems und dessen Attribute, so muss auch diese Änderung modelliert und betrachtet werden. Dazu dient, wie schon beschrieben, der Aufruf von Requires am Ende der Funktion. Eventuell ergeben sich durch den Fehler so weitere Konsequenzen für das Deployment, da die Anforderungen der Software oder des Geschäftsprozesses nicht mehr erfüllt werden können.

5.5.6 Funktionen auf der Ebene der IT-Infrastruktur

In diesem Abschnitt definieren wir die Funktionen, die in unserem Infrastrukturmodell IT-Systeme und Kommunikationskomponenten definieren. Dabei unterscheiden wir in unserem Modell zwischen aktiven Kommunikationskomponenten oder IT-Systemen.

Anlegen eines IT-Systems

Die Funktion creatITS dient zum Anlegen eines IT-System in unserem Infrastrukturmodell. Dabei kann eine Instanz angegeben werden.

Bezeichner [x Instanzbezeichner] → IT-System

createITS(„Bezeichner" [, „Instanzbezeichner"]) = ITS

createITS(„Server1") = ITS1, dabei gilt:

Sei ITS die Menge aller IT-Systeme,

$\forall\ ITS_a \in ITS: Bezeichner(ITS_a) \neq Bezeichner(ITS1)$

Formel 62 – Funktion zum Anlegen eines IT-Systems

Wie beschrieben werden wir in den weiteren Beispielen innerhalb der IT-Infrastrukturmodelle die Instanzen zur Modellierung von Endpunkten an IT-Systemen benutzen.

Anlegen einer aktiven Netzkomponente

Die Funktion creatNK dient zum Anlegen einer Netzkomponente in unserem Infrastrukturmodell. Dabei kann eine Instanz angegeben werden.

Bezeichner [x Instanzbezeichner] → Netzkomponente

createNK(„Bezeichner" [, „Instanzbezeichner"]) = NK

createITS(„Switch1") = NK1, dabei gilt:

Sei NK die Menge aller Netzwerkkomponenten,

$\forall\ NK_a \in NK: Bezeichner(NK) \neq Bezeichner(NK1)$

Formel 63 – Funktion zum Anlegen einer Netzwerkkomponente

Wie beschrieben werden wir in den weiteren Beispielen innerhalb der IT-Infrastrukturmodelle die Instanzen zur Modellierung von Endpunkten an Netzkomponenten benutzen.

Zuweisen von Attributen an ein IT-System oder eine Netzkomponente

Wie auch bei Softwarekomponenten definieren wir nur exemplarisch die Funktionen zur Zuweisung von Attributen an IT-Systeme oder Netzkomponenten. Diese entsprechen den Definitionen von Geschäftsprozessen.

ITS1.setAttrib(„arch“,“x64“)

NK1.setAttrib(“AnzahlPorts”,”8”)

Formel 64 – Funktionen zur Zuweisung von Attributen an ein IT-System/Netzkomponente

Hierarchische Anordnung

Wir definieren Hierarchien von IT-Systemen. Wir verzichten in unserem Infrastrukturmodell auf die Modellierung von Hierarchien auf aktiven Netzwerkkomponenten. Auf der Ebene der passiven Netzwerkkomponenten haben wir Hierarchien bereits durch die Kanäle definiert. Passive Netzwerkkomponenten sind hier immer die unterste Hierarchie des Kanals und realisieren die physikalische Übertragungsschicht.

Verfeinerung

Die Verfeinerung ordnet einem IT-System ein anderes IT-System unter. Dadurch entsteht eine Containerbeziehung zwischen den beiden IT-Systemen.

IT-System x IT-System → IT-System x IT-System x Hierarchiebeziehung

addsubITS(ITS1, ITS2)

Dabei gilt: $\forall$ $ITS_a \in$ *ITS gilt ischild(ITS1,* ITS_a*)=false oder* ITS_a *= ITS2*

Formel 65 – Funktionen zur Verfeinerung eines IT-Systems

Wir lassen in unserem Infrastrukturmodell nur eindeutige Hierarchien zu. Dies entspricht der Modellierung von Bäumen. Ein IT-System kann damit nicht in mehreren IT-Systemen existieren. Wir definieren auf den anderen Ebenen die Überprüfung der Hierarchischen Beziehung.

IT-System → IT-System

parent(ITS1) = ITS2

IT-System → { IT-System } *

children(ITS2) = ITS1

Formel 66 – Kindelemente einer Hierarchiebeziehung

IT-System x IT-System → Bool

ischild(ITS1, ITS2) = true, wenn gilt: ITS1 $\in$ *children (ITS2)*

ischild(ITS1, ITS2) = false, wenn gilt: ITS1 $\notin$ *children (ITS2)*

isparent(ITS1, ITS2) = false, wenn gilt: ITS1 $\notin$ *children (ITS2)*

isparent(ITS1, ITS2) = true, wenn gilt: ITS1 $\in$ *children (ITS2)*

isroot(ITS1) = true, wenn gilt:

Sei ITS die Menge aller IT-Systeme

$\forall$ $ITS_a \in$ *ITS gilt ischild(ITS1,* ITS_a*)=false und*

E $ITS_b \in$ *ITS mit ischild(*ITS_b*, ITS)=true*

isatomar(ITS1) =true, wenn gilt:

Sei ITS die Menge aller IT-Systeme

$\forall$ *ITS*$_a \in$ *ITS gilt isparent(ITS*$_a$ *,ITS1)=false*

Formel 67 – Funktionen auf Hierarchiebeziehungen

Binden eines Endpunktes an ein IT-System

Die Verbindung der IT-Systeme untereinander wird über Kanäle modelliert, die an Endpunkte gebunden sind. Endpunkte werden dabei entweder an ein IT-System oder an einen anderen Endpunkt gebunden. Die Funktionen zur Bindung haben wir bereits in Kapitel 5.1.5 angegeben.

Die Funktion Requires

Die Funktion Requires prüft, ob alle Anforderungen für die Installation einer Softwarekomponente auf einem IT-System erfüllt sind.

```
IT-System x Softwarekomponte → Bool
requires(IT-System i, Softwarekomponente s) {
    for each Requiremnet in s {
            if i.contains(s.Requirement) return else quit;
    }
   return true;
}
```

Formel 68 – Requires prüft die Anforderungen

Dabei müssen bei dynamischen Attributen und Anforderungen auch die entsprechenden Funktionen ausgewertet werden. Durch die Anbindung unseres Modells an ein Managementsystem entsteht die Möglichkeit, die Attribute aus dem Managementsystem zu verwenden und so zum Beispiel die reale Auslastung eines IT-Systems in die Modellierung zu integrieren. Wird eine Komponente auf einem System installiert, werden die Funktionen entsprechend an das System gebunden. Damit verringern sich die freien Ressourcen auf dem jeweiligen System. Dies wirkt sich auf die Attribute der jeweiligen IT-Komponenten aus, die dann mit dem Managementsystem abgeglichen werden können. Dabei können statische Attribute entsprechend verkleinert werden. Die Modellierung mit dynamischen Attributen ermöglicht die Modellierung einer Funktion, die aus den Requirements der installierten Softwarekomponenten die Restkapazität berechnet.

Abbildung von IT-Systemen auf Feature

Die Abbildung von IT-Systemen auf Features stellt die Verbindung zwischen den Features und den IT-Systemen her.

IT-System x Feature → Featurebindung

connectITF(IT-System, Feature) = Featurebindung

Formel 69 – Funktion zur Abbildung von Geschäftsprozessen auf Features

IT-Systeme führen Software aus, sie haben von sich aus keine allein stehende Funktion. Die Abbildung zwischen einem IT-System und einem Feature erfolgt deswegen meist beim Deployment von Software auf einem IT-System.

Ein Funktionsmodell kann jedoch auch feingranularer modelliert werden. So lässt sich zum Beispiel eine Funktion Datenhaltung modellieren, die dann mit den entsprechenden IT-Systemen der SAN Infrastruktur verbunden wird.

5.6 Softwarekomponenten auf IT-Systeme - Deployment

Die Abbildung von Softwarekomponenten und Applikationen auf IT-Systeme entspricht dem Deployment oder Installation der Software auf der realen Infrastruktur, die eine Beziehung zwischen dem IT-System und der Softwarekomponente festlegt.

> *Das Deployment beschreibt die Einrichtung einer Softwarekomponente auf einem IT-System, die Verfeinerung der angebotenen Schnittstellen an die Eigenschaften des IT-Systems und Bindung an einen (oder mehrere) Endpunkt(e) des IT-Systems. Dabei müssen die modellierten Anforderungen der Softwarekomponente erfüllt sein.*

Definition 35 – Deployment

Damit werden IT-Systeme über die Softwareebene mit den Geschäftsprozessen verbunden. Darüber hinaus werden den IT-Systemen so auch die Funktionen zugeordnet.

Wie in Kapitel 5.4 beschrieben, bestehen Anwendungen aus Softwarekomponenten. Darüber hinaus können Softwarekomponenten wieder hierarchisch unterteilt werden. Es sollten nur atomare Softwarekomponenten auf ein IT-System zugewiesen werden, wir erzwingen dies jedoch nicht, damit auch Containerbeziehungen wie Java-Runtimes modelliert werden können. Diese sind keine atomaren Softwarekomponenten, da sie andere Softwarekomponenten beinhalten und ausführen. Dieses Mittel sollte jedoch sparsam eingesetzt werden.

Wir definieren die Funktion zur Abbildung.

Softwarekomponente [Instanz] x IT-System x Endpunkt → Deploymentabbildung

connectSIT(Softwarekomponte | IT-System, IT-System, Endpunkt) = Deploymentabbildung

connectSIT(SK1, Server1, EP1)=Dep1

Dabei gilt:

$\forall\ SH_a \in SK1.provides$ *gilt:* $compat(SH_a, EP1) = true$

$\forall\ SH_a \in SK1.provides$ *gilt:* $Optionszahl(SH_a) = 0$

#EP1 >= #SK1.provides

requires(Server1, SK1) = true

Formel 70 – Funktion zur Abbildung von Softwarekomponenten auf IT-Systeme

Die Funktion setzt voraus, dass alle Schnittstellen, die eine Softwarekomponente zur Verfügung stellt, kompatibel mit dem Endpunkt sein, an den sie gebunden wird. Für den Endpunkt gilt, dass die Zahl der Schnittstellen nach dem Deployment mindestens der Zahl der Schnittstellen entspricht, die die neue hinzugefügte Softwarekomponente zur Verfügung stellt. Für diese Schnittstellen gilt dabei, dass alle Schnittstellen eindeutig sein müssen und keine Optionen mehr enthalten. Die Schnittstellenhierarchie wird entsprechend der Zuweisung an den oder die Endpunkte des IT-Systems verfeinert. Gleiches gilt für die in den Requ-Attributen bzw. Provides-Attributen modellierten Beziehungen der Softwarekomponenten, auch in diesen werden die konkreten Schnittstellen der IT-Systeme und Komponenten verfeinert, mit denen die jeweilige Softwarekomponente in der IT-Infrastruktur kommuniziert. Damit werden die Beziehungen, die für die Komponente gelten, auf die Infrastruktur abgebildet und von der abstrakten in eine konkrete Beziehung übergeführt.

Damit ist eine Abbildung der Softwarekomponenten auf das/die IT-Systeme festgelegt. Je nach Bedarf kann eine Softwarekomponente zur Steigerung der Verfügbarkeit oder zur Steigerung der Leistungsfähigkeit auch auf mehrere IT-Systeme installiert werden. In diesem Fall wird die Komponente dann an die jeweiligen Endpunkte der IT-Systeme gebunden und ist mehrfach vorhanden. Ein Spezialfall sind hier Softwarekomponenten, die auf Serverclustern ausgeführt werden. Diese werden nicht an die Endpunkte der physikalisch vorhandenen IT-Systeme ge-

bunden, sondern an den virtuellen Endpunkt des Clusters. Sie sind damit nicht direkt an die Clusterknoten gebunden, die Bindung erfolgt über den virtuellen Endpunkt, der an die physikalischen Endpunkte der Clusterknoten gebunden ist. Diese Abbildung entspricht der tatsächlichen Bereitstellung der Software auf den Knoten mit Techniken wie Microsoft Cluster Services.

Für eine vollständige Ausführbarkeit eines Geschäftsprozesses müssen alle Softwarekomponenten, bei denen eine Beziehung zum Geschäftsprozess besteht, mindestens an ein IT-System und einen Endpunkt gebunden werden. Es müssen alle Requirements des Geschäftsprozesses erfüllt sein. Alle Abhängigkeiten von Softwarekomponenten von anderen Komponenten müssen erfüllt sein. Dann ist der Geschäftsprozess, realisiert durch die Softwarekomponenten auf der IT-Infrastruktur, ausführbar.

5.7 Konsistenzsicherung

Die Anwendung der von uns beschriebenen Funktionen stellt die Konsistenz des Modells in einem weiten Rahmen sicher. Wir haben jedoch zwei Ausnahmen zugelassen, die zu einem unterspezifizierten Modell führen können.

Die erste Ausnahme ist, dass im Softwaremodell Schnittstellen hinzugefügt werden können, die mit dem Platzhalter * Operator nicht vollständig spezifiziert sind. Wenn das Modell in ein Tool integriert wird, kommt es hier auf das Anwendungsszenario an, ob diese Tatsache ein Problem darstellt. Wird das Modell für einen Managementkomponente verwendet, so muss diese Unterspezifikation aufgelöst werden. Dazu gibt es zwei Möglichkeiten:

- Alle nicht zugewiesenen Softwarekomponenten werden aus dem Modell entfernt. In diesem Fall gehen Informationen über modellierte Softwarekomponenten verloren.
- Die nicht zugewiesenen Softwarekomponenten bleiben erhalten und das Tool implementiert selbst eine Deployfunktion. Damit ist ein Managementwerkzeug in der Lage, Änderungen durch das Zuweisen von neuen Softwareverteilungen auf Server selbständig durchzuführen. Diese Realisierung ist für das in der Vision beschriebene Tool notwendig.

Dies setzt voraus, dass alle für die Geschäftsprozesse notwendigen Softwarekomponenten auch mindestens auf einem IT-System ausgeführt werden und alle Ablaufbeziehungen erfüllt werden können.

Das zweite Problem im Bezug auf die Konsistenz betrifft die Schnittstellenhierarchien unseres Modells. Wir haben aus Gründen der Vereinfachung der Formeln auf das Prüfen auf fest definierte Hierarchien innerhalb des Modells verzichtet. Die Funktionen stellen zwar sicher, dass keine Schnittstellen mit Kanälen verbunden werden, die nicht derselben Hierarchie entsprechen, doch wäre es für den Benutzer einfacher, wenn er bereits bei der Definition von Schnittstellenhierarchien die Hilfe hätte, dass nur bereits definierte Hierarchien verwendet würden. Dies würde gerade das Problem von Tippfehlern verringern und sollte in ein entsprechendes Tools durch die Auswahl vordefinierter Schnittstellenhierarchien, die vom Benutzer geändert werden können, integriert werden. Hier handelt es sich aber um eine Vereinfachung und keine Ungenauigkeit des Modells.

Ein weiterer Punkt ist die Prüfung von Anforderungen des Modells, die wir in die Funktionen zum einfacheren Verständnis nicht integriert haben. Diese Funktionen müssen alle Req Attribute des Modells entsprechend prüfen. Dazu muss zum Beispiel auch bei den Softwarekomponenten geprüft werden, ob die entsprechenden Kanäle existieren.

Wichtig ist die Integration der verschiedenen Requirements vor allem bei der Umsetzung des Modells in einem Tool. Gerade bei der manuellen Modellierung müssen die Requirements vom Modellierer entsprechend beachtet werden. Bei komplexen Modellen wird dies sehr schwierig und sollte nur noch unterstützt durch Tools durchgeführt werden.

5.8 Zeit innerhalb des Modells

Bei der Beschreibung unseres Models und der Teilmodelle sind wir nicht auf einen Zeitbegriff eingegangen. Dieser Abschnitt zeigt, wie sich Einschränkungen im Bezug auf die Zeit in den verschiedenen Teilmodellen beschreiben lassen. Dabei wird vor allem die Modellierung zeitlicher Einschränkungen von Geschäftsprozessen und ihre Integration in das Modell dargestellt. Ein absoluter Zeitbegriff für die Softwarekomponenten und die IT-Infrastruktur im Sinne einer Systemzeit oder der Modellierung von Systemtakten wird dabei nicht vorgenommen.

Gerade Betrachtungen und Optimierungen über die Zeit sind ein komplexes Gebiet, das nicht von unserer Arbeit abgedeckt wird. An vielen Modellen und Ansätzen wird immer noch geforscht. Es ist sehr schwierig für IT-Systeme absolute Abläufe anzugeben, da sich die Systeme durch ihre hochgradige Vernetzung und die sehr stark ausgeprägte Nebenläufigkeit von Betriebsystem und verschiedenen Anwendungssysteme auszeichnen. Die Einflüsse, die die Systeme dadurch aufeinander ausüben, sind nicht immer deutlich und gerade bei einer hohen Auslastung spielen Effekte wie Speichermanagement und die Rechnerarchitektur eine entscheidende Rolle. Neue Konzepte wie Virtualisierung verstärken diese Effekte noch. Wir geben in diesem Kapitel einen Einstieg, wie fachliche Anforderungen modelliert werden können, erlauben aber keine Modellierung des Systemverhaltens über die Zeit.

Im Gegensatz zu so genannten Echtzeitsystemen, die unter anderem häufig in eingebetteten Systemen eingesetzt werden, genügen betriebliche Informationssysteme meist nicht den harten zeitlichen Anforderungen, die für Echtzeitsysteme gelten. Diese garantieren unter anderem Reaktionsgeschwindigkeiten und Reaktionszeiten. Dies ist bei betrieblichen Informationssystemen durch deren deutlich größere Komplexität, sowie Effekte wie Nebenläufigkeit und Einflüsse durch verschiedene Abstraktionsstufen (Maschinenebene, Betriebsystem, Laufzeitumgebung, Anwendung) nicht möglich. Eine Planung und Steuerung einer entsprechenden Infrastruktur auf Sekunden genau, ist deshalb nicht möglich. Daraus ergibt sich auch, dass Abläufe in diesen Systemen gerade bei einer hohen Auslastung durch Nebenläufigkeitseffekt mit anderen Komponenten und konkurrierende Zugriffe auf gemeinsame Ressourcen nicht immer in der gleichen Zeit ausgeführt werden können. Gerade neue Laufzeitumgebungen wie die Java Runtime und .NET führen durch den Einsatz von asynchronen Systemprozessen wie dem Garbage Collector[79] dazu, dass absolute Abschätzungen der Laufzeit nicht möglich sind, vor allem nicht bei einer hohen Auslastung der Systeme. Auslagerungsprozesse vom Hauptspeicher in Hintergrundprozesse und die Einflüsse von mehrschichtigen Cachearchitekturen[80] machen Abschätzungen noch komplexer.

Diese Einschränkungen müssen bei der Modellierung beachtet werden. Unser Modell kann nur als Annäherung verwendet werden. Eines der größten Probleme, die Nebenläufigkeit von unabhängigen Prozessen und deren Konkurrieren um gemeinsame Ressourcen (Prozessor, Speicher,…), wodurch sie sich gegenseitig beeinflussen, kann dabei durch unser Modell verdeutlicht werden. Damit steht eine Möglichkeit zur Verfügung, Probleme aufzulösen indem Softwarekomponenten neu verteilt werden, oder Dienste während der Ausführung anderer Dienste abgeschaltet werden, so dies die fachlich modellierten Anforderungen des Geschäftsprozesses erlauben. Dadurch kann die Gesamtauslastung aller IT-Systeme über die Zeit gesteigert werden, ohne die Ausführungsbedingungen der Einzelsysteme zu beeinflussen.

---

[79] Die Aufgabe des Garbage Collectors ist es, nicht mehr benötigte Objekte aus dem Speicherbereich der Applikation zu entfernen. Die Ausführung des Garbage Collectors wird von der Laufzeitumgebung gesteuert. Dazu wird Rechenleistung benötigt, die der Applikation nicht zur Verfügung steht, die Steuerung, wann diese Leistung der Anwendung fehlt, wird aber nicht von der Anwendung selbst bestimmt. Damit kann die Anwendung nicht garantieren, bestimmte Zeitschranken einhalten zu können.

[80] In heutigen Serversystemen werden teilweise 4-Stufige Cachehierarchien verwendet, die jeweils eigene Optimierungsstrategien verfolgen, die nicht nur Performance sondern auch auf Stromverbrauch optimieren.

[Heg99] führt einen Zeitbegriff in Managementsystemen ein, der in einem Regelsystem die zeitlichen Anforderungen an eine Komponente definiert. Dieses Regelsystem beschreibt dabei nur die technische Sicht auf diesen Dienst. Mit unserem IT-Dienstmodell ist die Modellierung von zeitlichen Anforderungen nicht nur auf ein Regelsystem für eine einzige Komponente begrenzt. Wichtiger ist in diesem Zusammenhang für unser Modell die Möglichkeit, auch zeitliche Anforderungen des Geschäftsprozesses auf fachlicher Ebene modellieren zu können. Diese müssen entsprechend mit der technischen Ebene abgeglichen werden.

5.8.1 Zeit und ihre Bedeutung auf den unterschiedlichen Ebenen

Die Zeit hat auf den verschiedenen Ebenen des Modells unterschiedliche Einflüsse und Bedeutungen. Bei der Modellierung und beim Übergang zwischen den Ebenen müssen diese Unterschiede berücksichtig werden.

Geschäftsprozessebene

Auf der Ebene der Geschäftsprozesse ist die Zeit im Hinblick auf fachliche Anforderungen relevant. So können Geschäftsprozesse zeitlich eingeschränkt sein, dass sie bis zu einem bestimmten Zeitpunkt ausgeführt werden müssen, oder aber ihre Laufzeit begrenzt ist. Diese beiden Anforderungen lassen sich ineinander überführen, wenn die Startzeit beliebig ist.

Es ist jedoch auch möglich, dass sowohl die Start- als auch die Endzeit eines Geschäftsprozesses fachlich bestimmt sind (Siehe 5.8.3). Dadurch ergibt sich eine beschränkte Ausführungszeit, die dann maßgeblich für die Anforderungen an die IT-Infrastruktur werden kann. Diese muss die dem Geschäftsprozess zugeordneten Softwarekomponenten in der begrenzten Zeit ausführen können.

Wichtig ist diese Information also sowohl beim Deployment der Softwarekomponenten auf die IT-Systeme, da hier auf eine entsprechende Ausführbarkeit innerhalb der Zeitschranken geachtet werden muss, als auch für die Überwachung der Ausführung, die die Bedingungen überprüfen muss. Entsprechend müssen diese fachlichen Vorgaben auf die Softwarekomponenten übertragen werden und vom Managementsystem überwacht werden.

Softwareebene

Die Beziehungen zwischen dem Geschäftsprozessmodell und dem Softwaremodellen beschreiben, welche Softwarekomponenten zur Ausführung eines Geschäftsprozesses benötigt werden. Diese Beziehung ist statisch, sie wird bei der Implementierung und der Einführung der jeweiligen Softwarekomponenten auf der IT-Infrastruktur festgelegt. Eventuell vorhandene Möglichkeiten der Änderung der Software durch einen Geschäftsprozessverantwortlichen, die in den Softwarekomponenten vorhanden sein können und durch Techniken wie BPEL oder BPM ermöglicht werden, helfen die Anpassung von Software an neue Gegebenheiten zu beschleunigen, werden aber nicht häufig und nicht in Echtzeit durchgeführt[81]. Darüber hinaus werden hier häufig nur die Parameter von einzelnen Aktionen verändert und nicht die Geschäftsprozesse an sich. Der Einfluss der Zeit auf diese Abhängigkeit kann deshalb vernachlässigt werden.

Auf der Ebene der Softwarekomponenten ist für die Betrachtung von Zeit vor allem die Ausführungszeit von einzelnen Abläufen in den Komponenten wichtig. Diese wird neben der Software auch von der IT-Infrastruktur bestimmt. Deshalb lassen sich zeitliche Effekte nur zu einem Referenzsystem angeben. So können Abläufe und die entsprechende Ausführungszeit in einem Softwaresystem auf einem Referenzsystem analysiert und dann im Bezug auf dieses angegeben werden.

[81] Treten in einem Geschäftsprozess menschliche Akteure auf, die innerhalb des Geschäftsprozesses eine Rolle spielen und Einfluss auf den Geschäftsprozess haben, so ist dies eine Aktion innerhalb des Geschäftsprozesses und nicht zwischen den verschiedenen Ebenen. Ein entsprechender Akteur beeinflusst die Abhängigkeiten zwischen den Teilmodellen also nicht, sondern ist selbst ein Bestandteil des Prozesses.

Beim Deployment der Softwarekomponenten auf die IT-Infrastruktur sind diese Referenzwerte entscheidend um zu prüfen, ob die fachlichen Zeitanforderungen des Geschäftsprozesses von der IT-Infrastruktur erfüllt werden können[82].

In heutigen Rechenzentren müssen die Anforderungen entsprechend bei der Planung berücksichtigt werden. Die Informationen sollten auch entsprechend in die Monitoringwerkzeuge integriert werden, um die Einhaltung der Anforderungen zu überwachen.

IT-Infrastruktur

Auf der Ebene der IT-Infrastruktur betrachten wir Zeit, wie bereits beschrieben, nicht im Sinne von Echtzeit. Modellierte Anforderungen eines Geschäftsprozesses an die Ausführungszeit müssen beim Deployment entsprechend berücksichtig werden. Dabei werden die auf dem Referenzsystem gemessenen Abläufe auf das konkrete System übertragen. Häufig werden die Anforderungen des Geschäftsprozesses sich nicht auf eine Ausführung beziehen, sondern auf viele Ausführungen (Durchläufe). Kann ein einziges IT-System die geforderte Ausführungszeit nicht garantieren und ist der Prozess parallelisierbar, so können mehrere Systeme gemeinsam den Geschäftsprozess ausführen (Scale-out).

Die Anforderungen und die Lastprofile der Softwarekomponenten und ihrer Geschäftsprozesse verändern und erweitern das Auslastungsprofil des Zielsystems. Mit diesem Auslastungsprofil und den Anforderungen der Geschäftsprozesse lässt sich die Verteilung von Softwarekomponenten auf IT-Systeme planen und mehrere, zeitlich voneinander unabhängige Geschäftsprozesse, können auf derselben Plattform betrieben werden.

Die Zeit spielt auch bei Veränderungen an den Softwarekomponenten auf der IT-Infrastruktur eine Rolle. Wird eine Softwarekomponente von einem IT-System auf ein anderes verschoben, kann der Zeitpunkt dieses Vorgangs durch Abläufe innerhalb der Softwarekomponente oder Einschränkungen durch den Geschäftsprozess bestimmt werden. Wird zum Beispiel ein Geschäftsprozess ausgeführt, der nicht unterbrochen werden darf, kann eine Verlagerung der Softwarekomponente nicht erfolgen. Gleiches kann für andere Änderungen der Infrastruktur gelten, zum Beispiel Änderungen an der Kommunikationsinfrastruktur. Mithilfe des Modells lassen sich aber entsprechende Anforderungen analysieren und Wartungsfenster so gezielt planen, dass ein ununterbrochener Betrieb trotz Wartung möglich ist.

Funktionsmodell

Zeit hat keinen Einfluss auf das Funktionsmodell, da die Features die potentiellen Funktionen des Gesamtmodells beschreiben und sich diese nur eine Änderung am Gesamtmodell, aber nicht aus sich selbst heraus ändern können.

5.8.2 Zeit und Abhängigkeiten innerhalb der Ebenen

Die modellierten Beziehungen zwischen Softwarekomponenten über ihre Schnittstellen, die Bindung der einzelnen Softwarekomponenten an IT-Systeme und die damit festgelegte Bindung an die IT-Infrastruktur sind ohne einen Zeitbegriff modelliert. Wir verzichten auf die Modellierung eines Systemtaktes oder anderer zeitlicher Einschränkungen oder Synchronisationsmechanismen. Wir gehen davon aus, dass die Abhängigkeit immer besteht und nicht an ein Zeitintervall gebunden ist. Dies bezieht sich nicht auf Managementsysteme, die eine Umverteilung der Komponenten ermöglichen, die dazu ja genau die Funktionen der Teilmodelle zur Modellierung der Beziehungen verwenden. Diese zeitlichen Einflussfaktoren sind Veränderungen des Modells und keine zeitlichen Deploymentbeziehungen.

Ist für einen Ablauf die Gesamtausführungszeit wichtig, so muss diese Durchlaufszeit als Anforderung an die einzelnen Softwarekomponenten modelliert werden. Sinnvollerweise wird diese Anforderung an eine Gesamtanwendung modelliert, die alle verwendeten Hardwarekomponenten einschließt. Damit ist auch die verwendete Infrastruktur mit eingeschlossen. Der Ablauf ist

[82] Wir geben kein Modell zur Übertragung der Ergebnisse an und betrachten dies als ein eigenständiges Forschungsfeld.

dann durch seinen jeweiligen Start- und Endpunkt definiert. Entsprechende Anforderungen müssen bei der Installation berücksichtigt werden. Wichtig ist hierbei, ein entsprechendes Referenzmodell anzugeben, nach dem der Ablauf gemessen wird.

Bei einer einfachen Abhängigkeit kann die Modellierung über dynamische Anforderungen und entsprechende Querbezüge in den Funktionen möglich sein.

Die Modellierung von Zeit ist gerade für die Planung und Optimierung der Auslastung sowie des Laufzeitverhalten der IT-Infrastruktur wichtig. Wir haben beschrieben, wie sich fachliche Anforderungen in das Modell einbeziehen lassen und so bei der Planung mit betrachtet werden können. Eine kontinuierliche Optimierung ist deutlich schwieriger zu erreichen und nicht Teil dieser Arbeit. Gerade durch die Einflüsse von Virtualisierung entstehen hier zusätzliche Probleme in der Planung. Wir betrachten dies als eigenständiges Forschungsgebiet.

5.8.3 Bewertungen anhand von Zeit

Für die Bewertung von Ausfällen kann es notwendig werden, die Zeit zu betrachten. Die im vorigen Abschnitt beschriebenen Funktionen geben an, wie sich ein Ausfall auf die verschiedenen Komponenten auswirkt. Dabei sind zeitliche Einflüsse nicht betrachtet. Dies ist vor allem bei asynchronen Vorgängen schwierig und kann zu unterschiedlichen Ergebnissen führen[83]. Wird zum Beispiel ein Produktionssystem Nachts mit Daten versorgt, die im Laufe des Tages abgearbeitet werden, ist der Ausfall der Kommunikation zu bestimmten Zeiten tolerierbar.

Sollen entsprechende Systeme modelliert und die Auswirkungen von Ausfällen bewertet werden, müssen die Ausfallfunktionen entsprechend um zeitliche Informationen erweitert werden. Wir veranschaulichen dies anhand eines Beispiels, das wir später im Rahmen der Evaluierung unseres Models erneut aufgreifen werden.

Bei einem Dienstleister werden am Monatsende alle erbrachten Dienstleistungen verrechnet und für die jeweiligen Kunden zusammengestellt. Daraus werden Rechnungen erstellt, die dem Kunden zu einem Stichtag zugestellt werden müssen. Für diesen Vorgang steht nur ein begrenzter Zeitraum zur Verfügung. Kann die Rechnung nicht erstellt werden, ist eine Rechnungsstellung erst im nächsten Abrechnungszeitraum möglich. Wir gehen davon aus, dass das System nur am Monatsende ausgeführt wird. Treten in dieser Zeit Fehler auf, ergibt sich der verlorene Ertrag aus der Dauer des Ausfalls und den zu erstellenden Rechnungen. Die Logik, welche Rechnungen in diesem Fall nicht erstellt werden, ist in der Funktion calcloss verborgen. Die Funktion faillostmoney wäre dabei an die fail-Funktion des Geschäftsprozesses gebunden und würde ausgelöst, wenn ein Fehler in der Hierarchie der Fail-Funktionen nicht weiter abgefangen werden könnte und bis zum Geschäftsprozess durchgegeben würde.

[83] Bei synchronen Kommunikationsvorgängen ohne Puffer ist das Impact Management deutlich einfacher, da bei einem Ausfall der Kommunikation sofort der Fehlerfall eintritt. Dieser kann dann entsprechend bewertet werden.

```
fail () {
  faillostmoney(durationfail);
}
faillostmoney(durationfail) {
    if dayofmonth < 28 {
      return 0;
    }
    else {
      return(calcloss(durationfail,bills));
    }
}
```

Formel 71 – Fail mit modellierter Zeitabhängigkeit

Die Formulierung der Function calcloss ist dabei sehr stark abhängig von der internen Struktur der Software und kann möglicherweise nur vom Entwickler der Software angegeben werden und muss auf die entsprechende Hardware zugeschnitten werden.

5.8.4 Rahmenbedingungen für zeitliche Betrachtungen

Neben der Schwierigkeit der Modellierung des Softwareverhaltens existiert eine weitere Schwierigkeit: Lässt sich der Verlust über einen bestimmten Zeitraum eines Geschäftsprozesses überhaupt genau beziffern. Wir möchten diesen Punkt an einem Beispiel verdeutlichten:

Eine Produktionsstrasse stellt ein Produkt A her. Diese Produktionsstrasse produziert entweder zu 100% Auslastung oder steht stillt. In diesem Fall lässt sich der Verlust bei Ausfall für zehn Minuten angeben, indem berechnet wird, wie viele Produkte A in dieser Zeit nicht hergestellt wurden. Was passiert aber, wenn diese Reaktion aufgeweicht wird, die Strasse auch zu 50% produzieren kann. Noch schwieriger wird die Betrachtung, wenn das Produkt A ein Vorprodukt für mehrere andere Produkte ist und auch diese entsprechend nicht produziert werden können. Dieses Beispiel verdeutlicht, dass entsprechende Betrachtungen für IT-Dienste noch deutlich schwieriger sind. So lässt sich bei einem Webshop zum Beispiel angeben, welchen Umsatz dieser im Durchschnitt in zehn Minuten erwirtschaftet. Fällt er aus, lässt sich der Schaden mit diesem Mittelwert angeben. Die Bestimmung des absoluten wirtschaftlichen Schadens ist so aber nicht feststellbar. Eventuell warten einige Besteller einfach die zehn Minuten ab und bestellen danach, oder aber der Ausfall fällt in eine Marketingkampagne, bei der die Durchschnittswerte nicht verwendet werden können und ein deutlicher Imageschaden entsteht.

Wir betrachten eine Betrachtung zeitlicher Einflüsse für wichtig, diese sind jedoch nicht Bestandteil dieser Arbeit.

6 Erweiterungen und Anpassung des Modells

Wir haben in den vorigen Kapiteln unser Modell vorgestellt. Das Modell kann auch als eine Basis für Erweiterungen verwendet werden oder aber an die Bedürfnisse des jeweiligen Szenarios angepasst werden. Wir werden in diesem Abschnitt ein Beispiel für die Erweiterung des Modells angeben, mit dem Firewallregeln mit dem Modell beschrieben und ihre Einflüsse analysiert werden.

Des Weiteren gehen wir die Anpassung des Modells durch ein Tailoring ein. Dabei wird der Umfang des Modells entsprechend den Anforderungen des Szenarios angepasst.

6.1 Modellierung einer Firewall

Unser Modell gibt einen Rahmen vor, der als Basis für Erweiterungen dienen kann. Wir werden am Beispiel von Firewallregeln verdeutlichen, wie diese Funktionen aussehen und sich das Modell entsprechend anpassen lässt. Dies ist nur ein Beispiel für eine Erweiterung.

Die Integration von Firewallregeln ermöglicht es, die Wirkung der Firewall zu testen. Firewalls arbeiten nach dem Prinzip, die Angriffsfläche auf einen Dienst zu minimieren. Sie können dazu auf verschiedenen Ebenen ansetzen. Sie schränken die Zugriffsmöglichkeiten auf einen Dienst so ein, dass nur noch berechtigte Dienste auf die Schnittstellen und Endpunkte zugreifen können, die sie wirklich benötigen. Alle nicht benötigten Schnittstellen werden gesperrt. Wichtig ist dabei, dass die Firewall keine Verbindungen unterbricht, die andere Dienste für ein korrektes Arbeiten benötigen. Durch das Modell kann genau dies geprüft werden.

6.1.1 Szenario

Firewalls schränken den Zugriff auf IT-Systeme ein, um deren Sicherheit zu verbessern. Dazu wird entweder auf Layer 2/3 bereits der Zugriff auf Ports beschränkt, oder die Firewall versteht das Applikationsprotokoll und filtert die Anfragen aus diesem Datenstrom heraus. Beide Arten von Firewalls wirken im Sinne unseres Modells auf die Schnittstellen and den Endpunkt der jeweiligen Systeme.

6.1.2 TCP/IP basierte Firewalls

TCP/IP basierte Firewalls filtern Datenpakete auf der Ebene ihrer Protokolle und reichen damit von Layer 3 bis Layer 4 im TCP/IP Stack[84]. Sie setzen damit an den Netzwerkkarten des jeweiligen Servers an und definieren, welche Zugriffe möglich und welche verboten sind.

In unserem Modell finden sich diese Informationen im Infrastrukturmodell und den Schnittstellenhierarchien. Die Wirkweise der Firewallregeln lassen sich in den Schnittstellenhierarchien beschreiben und ihre Auswirkungen modellieren. Firewallregeln wirken auf die an den Endpunkt gebundenen Schnittstellen und können die Zahl der Schnittstellen des Endpunktes einschränken. Wir definieren, dass Firewallregeln auf den internen Endpunkt eines IT-Systems nicht angewendet werden können[85].

fwregel=allow(ethernet:ip[192.168.6.100]:tcp[80]:) and deny(*)*

firewall(E,fwregel)=E

Formel 72 – Firewallregel für einen Webserver auf Layer 3

[84] Einige Firewalls, wie der Linux Kernel, können hier auch von Layer 4 bis hinab zu Layer 2 filtern.

[85] Die Definition entspricht dem Verhalten gängiger Betriebsysteme wie Windows und MacOS. Linux kann entsprechende Firewallregeln verwalten. Eine Einschränkung der localhost Schnittstelle ist aber meist nur in virtualisierten Umgebungen auf dem Hostsystem sinnvoll.

Formel 72 definiert eine mögliche Firewallkonfiguration für einen Webserver. Die Regel löscht alle Schnittstellen vom Endpunkt E, bis auf die Regel für den Webserver und etwaig vorhandene Verfeinerungen dieser Regel. Hier sind die jeweiligen Instanzen der Protokolle auf dem Server definiert. Für IP die Adresse des Rechners, für das tcp Protokoll der Port 80. Die fehlende Instanz bei ethernet entspricht dem Verhalten der meisten Firewalls, erst auf Ebene 3 einzugreifen und Pakete unterhalb dieser Ebene nicht zu betrachten[86].

6.1.3 Layer 7 Firewalls

Firewalls können aber auch auf höheren Ebenen eingesetzt werden. Layer 7 Firewalls filtern den Datenverkehr, der über die Firewall abgewickelt wird auch inhaltlich. Diese Firewalls verstehen das verwendete Protokoll und können so eine Schnittstelle nicht nur ein- oder ausschalten, sondern auch kritische Funktionen der Schnittstelle deaktivieren. Auch diese Funktion lässt sich mit den Mitteln unseres Modells ausdrücken und die entsprechenden Funktionen integrieren.

Eine Firewal auf Ebene 7 blockiert im Sinne unseres Modells einzelne Schnittstellen auf einer höheren Hierarchie. Dies kann auf der Ebene eines Anwendungsprotokolls wir http erfolgen, aber auch auf einer noch höheren Ebene, also zum Beispiel der Schnittstelle, die ein Webservice implementiert.

fwregel=allow(ethernet:ip[192.168.6.100]:tcp[80]:http[get index.html]) and deny(*)

firewall(E,fwregel)=E

Formel 73 – Firewallregel für einen Webserver auf Layer 7

Die Formel 73 erweitert die Formel 72 um die zusätzliche Hierarchie http. Mit dieser Regel wird nur die http Methode get für die Datei index.html freigeschalten. Alle anderen Anfragen und bereitgestellten Schnittstellen durch den Webserver werden durch die Firewallregel verboten und aus dem Schnittstellenraum des Servers entfernt. Damit werden die Zugriffsmöglichkeiten auf den Server nochmals deutlich eingeschränkt, da der Service zwar noch verfügbar ist, aber nur sehr wenige Zugriffe erlaubt sind.

Wir zeigen in einem Beispiel in Kapitel 9.2.1, wie sich unser IT-Dienstmodell hier auch dazu verwenden lässt, Informationen aus der Entwicklung einer Softwarekomponente in den Betrieb zu übernehmen.

6.1.4 Funktionen zur Integration von Firewalls in das Modell

Die Integration der Firewallregeln in unser Modell erfolgt, in dem die Regeln an die Endpunkte der jeweiligen Systeme angewendet und anschließend als Attribut modelliert werden. Bei jedem neuen Deployment an diesem Endpunkt werden die Firewallregeln erneut nach der Abbildung der Schnittstellen angewendet.

Die Nutzung lässt sich erweitern, in dem die Regeln nicht direkt angewendet werden, sondern nur die Auswirkungen durchgespielt werden. So lassen sich die Abhängigkeiten zwischen Softwarekomponenten und IT-Systemen, die durch die Anwendung der Firewallregeln beeinträchtigt werden herausfinden und so die Auswirkungen der Einführung der Firewallregeln bestimmen, weil in diesem Fall die von den verschiedenen Softwarekomponenten benötigten Schnittstellen nicht mehr zur Verfügung stehen.

---

[86] Auch Serverprodukte verzichten teilweise auf einer Filterung auf Layer 2, da das Einschleusen von Datenpaketen auf dieser Ebene über die physikalischen Grenzen von Netzen nicht möglich ist. Innerhalb der physikalischen Netzwerkstruktur ist ein Angriff denkbar. Kritisch ist dieser Umstand vor allem bei WLAN Netzen, da hier der physikalische Zugriff nicht begrenzt ist. Deswegen können hier durch Treiberfehler und die Tatsache, dass Teile der Funktionen des Layer 2 auch in Software ausgeführt werden, Sicherheitsprobleme entstehen und der Zugriff auf einen Rechner an der Firewall vorbei möglich sein.

Prüft ein Modellierungswerkzeug die Anforderungen der Softwarekomponenten und hat eine Firewallregel eine notwendige Schnittstelle an einem Endpunkt deaktiviert, ist dieser Fehler so sehr leicht zu erkennen.

6.2 Tailoring

Neben der Erweiterung des Modells um neue Funktionen ist auch die Anpassung des Modells an verschiedenen Szenarien wichtig. Implizit beinhaltet unser Modell bereits Möglichkeiten zur Anpassung, durch die freie Wahl der Schnittstellen- und Kanalhierarchien ist es dem Anwender überlassen, wie detailliert diese beiden Aspekte modelliert werden.

Wir haben in den einzelnen Kapiteln auch bereits verschiedene Möglichkeiten der Anpassung angedeutet. In diesem Abschnitt gehen wir auf die Anpassung des Modells, bzw. auf ein Tailoring des Modells ein. Wir stellen eine Vereinfachung des Modells vor und verdeutlichen so die Möglichkeiten der Anpassung des Modells an die jeweiligen Anforderungen an die Modellierungstiefe.

ViSEK definiert das Tailoring als:

> *Tailoring bezeichnet das Anpassen eines generischen Objekts (z. B. einer Dokumentenvorlage oder eines Vorgehensmodells) an spezifische Bedürfnisse (z. B. einer Organisation oder eines Projekts).*
>
> *Bei allen Veränderungen ist darauf zu achten, dass die eigentliche Funktion des dem Tailoring unterzogenen Objekts erhalten bleibt. Daher werden häufig Regeln festgelegt, an denen sich das Tailoring zu orientieren hat.*

Definition 36 – Tailoring (nach ViSEK)

Wir geben hier die Regeln und das Vorgehen für dieses Tailoring an, um unser IT-Dienstmodell zu vereinfachen. Diese Variante des Modells ist vor allem dann interessant, wenn sehr einfache Geschäftsprozessmodelle und eine überschaubare Anzahl an Applikationen modelliert werden sollen. Dieses Modell ist deswegen vor allem für kleinere Unternehmen interessant, die Ihre Geschäftsprozesse nur in wenigen oder nur einer Ebene modellieren.

Im vereinfachten Modell werden die Geschäftsprozesse mit in den Schnittstellen modelliert. Es wird eine zusätzliche Schnittstellenhierarchie eingeführt, die die Geschäftsprozesse direkt darstellt. Dadurch wird der Schnittstellenstack zwar komplizierter, die Modellierung von Beziehungen zwischen dem Softwaremodell und dem Geschäftsprozessmodell entfällt jedoch. Zu beachten ist hier, dass unsere Erweiterungen zur Modellierung von Firewalls in diesem Szenario nicht mehr direkt angewendet werden können, sondern adaptiert werden müssen. Die technische Firewallregel schränkt dabei nicht den Geschäftsprozess ein, dessen Beziehungen bleiben erhalten.

Die Kanalhierarchie entfällt. Wir bilden alle Schnittstellenhierarchien auf einfache Kanäle ab. Damit entfällt die Möglichkeit der Partitionierung von Netzwerken auf einer logischen Ebene, die Kanäle sind aber sehr einfach modellierbar und entsprechen dem, was der Administrator „sieht“, der physikalischen Hardware, mit der er die verschiedenen IT-Systeme verbindet[87].

Darüber hinaus entfallen im vereinfachten Modell die Anwendungsobjekte innerhalb des Softwaremodells. Die Softwarekomponenten werden ohne die Klammerfunktion der Anwendungen modelliert[88].

---

[87] Damit entfällt auch die Modellierung der Abbildung von VLAN’s und virtuellen Netzwerkinfrastrukturen in einer Virtualisierungsplattform auf die zugrunde liegenden physikalischen Netzwerkinfrastrukturen.

[88] Da wir Softwarekomponenten als atomare Einheit definiert haben, ist es notwendig, diese zu modellieren und nicht die Anwendungen, aus denen diese aufgebaut sind. In einem für das vereinfachte Modell passenden Szenario werden die Softwarekomponenten aber meist mit den Anwendungen zusammenfallen.

Auch auf das Funktionsmodell verzichten wir im vereinfachten Modell. Da in diesem Szenario nur eine sehr begrenzte Zahl an Softwarekomponenten existiert, ist die Strukturierung nach den in den Schnittstellen modellierten Geschäftsprozessen sinnvoller. Ansonsten werden im Funktionsmodell nur die Funktionen jeweils einer Komponenten modelliert. Diese Informationen können auch als Attribute der einzelnen Softwarekomponenten modelliert werden.

	Einschränkung im vereinfachten Modell
Geschäftsprozessmodell	Das Geschäftsprozessmodell bleibt vollständig erhalten. Es werden keine Beziehungen zwischen Geschäftsprozessen und Softwarekomponenten modelliert. Die Geschäftsprozesse bilden die oberste Schnittstellenhierarchie
Softwaremodell	Das Softwaremodell enthält keine Anwendungen. Die Abbildung zwischen den Softwarekomponenten und den Geschäftsprozessen erfolgt durch eine zusätzliche Hierarchie in den Schnittstellen
Infrastrukturmodell	Das Infrastrukturmodell bleibt unverändert
Funktionsmodell	Das Funktionsmodell entfällt
Kanäle	Die Kanäle werden nicht in Hierarchien angeordnet. Die Existenz eines Kanal erlaubt die Kommunikation aller Schnittstellenhierarchien

Vorteile dieser Tailoringstufe:

- Die Erstellung von einfachen Modellen wird vereinfacht, eine formale Abbildung zwischen den Teilmodellen Geschäftsprozessmodell und Softwaremodell kann entfallen.
- Die Modellierung der Features der einzelnen Softwarekomponenten kann entfallen.
- Die Modellierung der Netzwerkinfrastruktur entspricht der physikalischen Struktur des Netzwerkes, logische Partitionierung wird nicht zugelassen.

Nachteile:

- Das Modell ist zwar in ein vollständiges Modell überführbar, an dieser Stelle müssen jedoch die verschiedenen Features modelliert werden. Die Schnittstellenhierarchien mit den Geschäftsprozessen müssen in entsprechende Beziehungen zwischen den Teilmodellen überführt werden. Die Kanalhierarchien müssen eingeführt werden. Das Funktionsmodell muss eingeführt und die Beziehungen entsprechend modelliert werden.
- Die Struktur und die Sicht des Funktionsmodells fehlt.
- Die Möglichkeit der Modellierung von Partitionen auf einer logischen Ebene entfällt durch den Verzicht auf die Kanalhierarchien. Teilnetze, die sich von physikalischen Netzen unterscheiden, sind in diesem Fall nicht modellierbar.

Es ist hier jedoch auch eine Variante denkbar, in der auch die Features modelliert werden.

6.3 Analyse der IT-Infrastruktur

Eine interessante Möglichkeit mit einem Modell der IT-Infrastrukutur und den darauf aufgeführten Softwarekomponenten ist die Analyse dieser im Bezug auf Fehlertoleranz. Eine wichtige Fragestellung ist, neben der Frage welche Auswirkungen ein Ausfall einer einzelnen Komponente hat auch die Frage, inwieweit eine IT-Infrastruktur auch in einem Fehlerfall noch ausgeführt werden kann.

Festplattensysteme in Servern sind dafür ein Beispiel. Entsprechende Systeme sind, wenn richtig konfiguriert, mit mehreren Platten zu einem Verbund organisiert. Man spricht in diesem Zusammenhang von RAID Systemen, einem redundant array of inexpensive/independant disks.

Der Unterschied zwischen inexpensive und independant ist historisch, beide Aspekte sind aber bis heute gültig. Innerhalb dieses Verbundes werden Daten mehrfach gespeichert (RAID1), die Dateninhalte werden dabei gespiegelt, oder aber Prüfsummen über die Datenanteile angelegt (RAID5), mit denen im Fehlerfall die Daten einer ausgefallenen Platte rekonstruiert werden können. Es existieren weitere RAID-Level, mit denen die Ausfallsicherheit weiter gesteigert werden kann. Fällt eine Festplatte aus, arbeitet das Gesamtsystem (eventuell mit verringerter Geschwindigkeit) weiter. Erst bei einem Ausfall einer zweiten Festplatte gehen Daten verloren (je nach RAID-Level und Konfiguration sogar erst nach dem Ausfall der dritten oder vierten Platte). Das System hat im Falle eines Ausfalls noch Reserven.

Das Wissen über diese Reserven ist wichtig bei der Risikoanalyse eines IT-Systems. Dabei ist die Suche eines Single-Point-of-failure (SPoF) eine wichtige Herangehensweise. Existiert in einer IT-Architektur eine Komponente, deren Ausfall das Gesamtsystem ausfallen lässt und deren Funktion nicht von einer zweiten Komponente erbracht werden kann, so spricht man von dieser Komponente als einem Single-Point-of-failure, da der Ausfall dieser Komponente nicht kompensiert werden kann.

Die Abbildung 65 verdeutlicht dies an einem Beispiel. Auf der obersten Ebene existieren zwei Webserver, die gemeinsam eine Website zur Verfügung stellen. Sie sind über zwei unabhängige Ethernetnetzwerke, die durch zwei Ethernet-Switche realisiert werden, mit zwei Datenbankservern verbunden, die eine Datenbank als Clusterdienst realisieren. Alle Server haben jeweils zwei Netzwerkkarten für die Nutzdaten, die an die verschiedenen Switche angeschlossen sind. Die Datenbankserver sind über zwei Fibrechannel-Switche mit einem gemeinsamen Speichersystem verbunden. Auch diese Pfade zum Speichersubsystem sind wieder redundant ausgelegt. Analysiert man dieses Szenario, stellt man fest, dass lediglich das Speichersystem einem Single-Point-of-Failure darstellt, alle anderen Komponenten sind doppelt ausgelegt.

Unser Modell kann diese Suche unterstützten. Auf der Ebene der Dienste an sich lässt sich über die Schnittstellenhierarchie sehr einfach herausfinden, ob ein Dienst mehrfach vorhanden ist, dies ist dann der Fall, wenn die Schnittstelle der Softwarekomponente auf mehreren Servern an einen jeweiligen Endpunkt gebunden ist, der einen Kanal zum Ausgangspunkt der Analyse hat. Dies ist wichtig, da sonst im Falle des Ausfalls für den analysierten Client trotzdem kein Dienst zur Verfügung steht. Schwieriger ist die Behandlung in diesem Fall, wenn Clustersysteme modelliert sind. Diese lassen sich in unserem Modell jedoch auch einfach erkennen. Alle Schnittstellen, die über eine Endpunktbindung nicht an ein IT-System sondern an einen anderen Endpunkt gebunden sind, müssen je nach der Fail() Funktion dieses Endpunktes betrachtet werden und sind, im Rahmen der Definitionen dieser Arbeit, hochverfügbare Dienste die von mehreren Servern erbracht werden.

Wichtig neben der Frage, ob die Dienste an sich ausgeführt werden ist jedoch auch die Frage, ob Verbindung zwischen zwei Servern existiert und diese Verbindung redundant ist. Diese Frage lässt sich ebenfalls mit dem Modell beantworten. Die logischen Verbindungen unseres Modells werden in den Kanalhierarchien auf die physikalischen Verbindungen abgebildet. Existiert ein physikalisch redundanter Pfad zwischen zwei Serversystemen, so werden auch die logischen Verbindungen zwischen diesen beiden Systemen auf die beiden zugrunde liegenden physikalischen Verbindungen abgebildet[89].

Das im vorigen Kapitel beschrieben Verfahren zur Modellierung von Firewallsregeln ist mit diesem Verfahren kompatibel, da die Firewallregeln an den Endpunkten greifen. Diese Überprüfung ist notwendig, da die Konfiguration der Firewallregeln an zwei Endpunkten unterschiedlich ausfallen kann und so die physikalisch redundante Verbindung durch die Friewallregeln wieder ausgehebelt wird.

Neben der Betrachtung, ob eine Verbindung zwischen zwei Systemen redundant ist, ist es mit dem Modell auch möglich, gezielt nach SPoF zu suchen. Auf der Ebene der Dienste ist diese

[89] Bei den schon erwähnten Verfahren Intel Switch Fault Tolerance und 802.3ad werden dazu auch im Betriebsystem des jeweiligen Servers virtuelle Netzwerkkarten eingebunden, die die redundante Verbindung im Betriebsystem als physikalische Verbindung repräsentieren.

Suche sehr einfach möglich. Es werden alle Dienste analysiert und überprüft, ob die Dienste auch auf einem anderen IT-System verfügbar[90] sind. Wichtig ist in diesem Zusammenhang wieder, dass die jeweilige Perspektive betrachtet werden muss. Die Ausführung eines Dienstes auf einem Server, der für den Client nicht erreichbar ist, leistet keinen Beitrag zur Steigerung der Verfügbarkeit und ist deshalb nicht sinnvoll. Die Konzepte des End-to-End Monitorings können hier verwendet werden und die Verfügbarkeit ausgehend vom Kundensystem analysiert werden. Dazu müssen gegebenenfalls verschiedene Perspektiven für verschiedene Kunden und Systeme betrachtet werden.

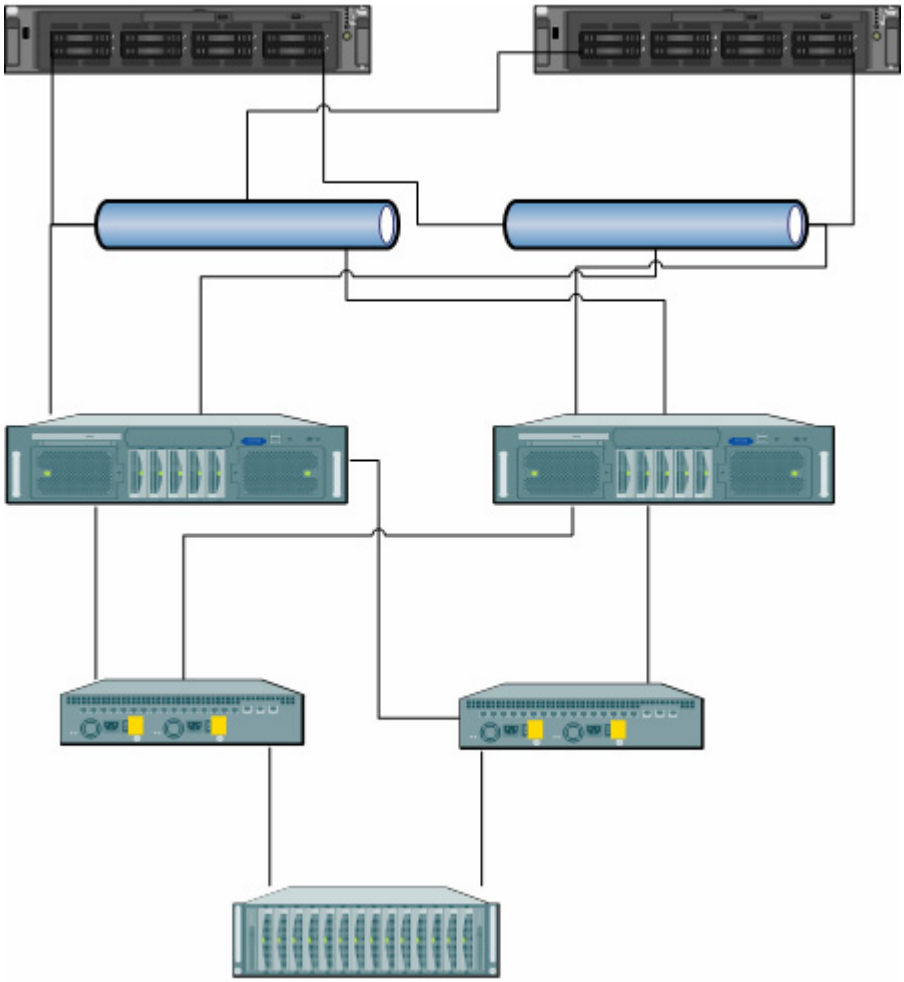

Abbildung 65 – HA Webfarm mit HA Datenbank

Schwieriger ist die Suche auf der Eben der Netzwerkinfrastruktur. Da wir unser Modell auf der Graphentheorie aufgebaut haben, können wir hier aber entsprechende Algorithmen verwenden. Abbildung 66 ist die grafische Darstellung der HA Webfarm mit der HA Datenbank. Die hellgrünen Knoten stellen dabei die IT-Systeme dar, dunkelgrüne Knoten die vier Netzwerkswitche. Auf die Modellierung des Clusters wurde hier zur Vereinfachung verzichtet, es wurden nur die physikalischen Verbindungen modelliert.

Die Suche nach SPoFs in der Netzwerkinfrastruktur lässt sich auf die physikalischen Verbindungen zurückführen. Existiert ein redundanter Netzwerkpfad zwischen zwei IT-Systemen und

[90] Die doppelte Ausführung eines Dienstes auf einem Server erhöht die Redundanz des Dienstes nicht, fällt der Server aus sind beide Instanzen des Dienstes betroffen.

wird über diesen Kanal eine Kanalhierarchie aufgebaut, so werden alle übergeordneten Kanalhierarchien auf die beiden physikalischen Kanäle abgebildet. Auf den höheren Ebenen können keine zusätzlichen Redundanzen entstehen, die nicht auf redundanten physikalischen Kanälen basieren. Ein spezieller Aspekt hierbei ist, dass die Verkabelung zwischen zwei Systemen redundant sein kann, durch die nächste Kanalhierarchie aber in zwei logische Verbindungen aufgespalten wird. Die Existenz eines redundanten physikalischen Pfades bedeutet nicht, dass die logische Verbindung automatisch auch redundant ist. Dies muss gesondert geprüft werden, indem für die höheren Kanalhierarchien geprüft wird, ob sie auf beide physikalischen Kanäle abgebildet sind.

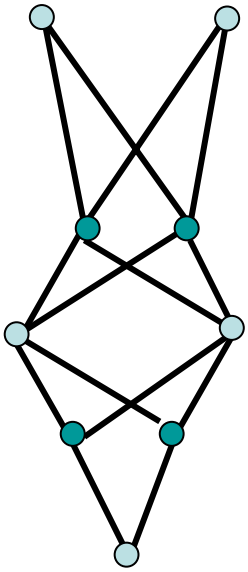

Abbildung 66 – Modell des HA Szenarios

Auf der physikalischen Ebene lässt sich die Suche nach einem SPoF darauf zurückführen, dass es eine Kante zwischen zwei IT-Systemen gibt, die beim Entfernen zu zwei unabhängigen Teilbäumen führt. Zur Suche kann das Konzept des Spannbaums verwendet werden. Ein Spannbaum entsteht aus einem ungerichteten Graphen, er beinhaltet alle Knoten des ursprünglichen Graphen und verbindet alle Knoten zu einem Baum. Für die Berechnung eines Spannbaums stehen verschiedene Verfahren zur Verfügung, unter anderem der Algorithmus von Kruskal und der Algorithmus von Prim. Ein anderer Weg wäre die in Graphen definierte Zusammenhangskomponente, für uns gilt jedoch eine Einschränkung:

Die Knoten in unserem Infrastrukturmodell unterscheiden sich in IT-Systeme Netzwerkkomponenten. Wichtig für die Frage, ob alle IT-Systeme miteinander redundant verbunden sind sind alle Knoten, die IT-Systeme beschreiben. Knoten, die Netzwerkkomponenten wie Switche beschreiben, müssen nicht in beiden Spannbäumen vorkommen.

Wir kommen somit zu einem Algorithmus, der einen minimalen Spannbaum der physikalischen Kanäle unseres Modells sucht. Dazu wird ein minimaler Spannbaum gesucht, der alle Knoten beinhaltet, die IT-Systeme darstellen. Alle Knoten, die Netzwerkkomponenten realisieren, und Teil des Spannbaums sind, werden im zweiten Schritt aus dem Modell entfernt, da die redundante Verbindung sich auch auf einer unabhängigen Netzinfrastruktur abstützen muss. Existiert dieser minimale Spannbaum nicht, existieren in der Netzinfrastruktur SPoF. Nachdem der erste Spannbaum gefunden ist und die entsprechenden Netzkomponenten aus dem Graphen entfernt, wird ein zweiter minimaler Spannbaum gesucht, der alle IT-Systeme beinhaltet. Existiert auch dieser zweite Spannbaum, dann sind alle IT-Systeme redundant aneinander angebunden und es existiert in der Netzinfrastruktur kein SPoF. Nun werden die Schnittstellenhierarchien der beiden Spannbäume verglichen. Jedes IT-System existiert in beiden Spannbäumen. Die Menge der Schnittstellen pro IT-System muss in beiden Spannbäumen an den Endpunkten identisch sein, ansonsten haben wir einen SPoF durch einen Fehler in der Konfiguration gefunden.

Redundante Pfade stellen mit all ihren Komponenten unabhängige Kanten dar. Der Algorithmus kann auch in Hochverfügbarkeitsszenarien eingesetzt werden. Wird ein Switch eingesetzt, der in sich selbst redundant ist, dann wird dieser in unserem Modell wie beschrieben auf zwei IT-Systeme zurückgeführt. Der Algorithmus trifft in diesem Fall auf zwei unabhängige Switche und erkennt keinen SPoF.

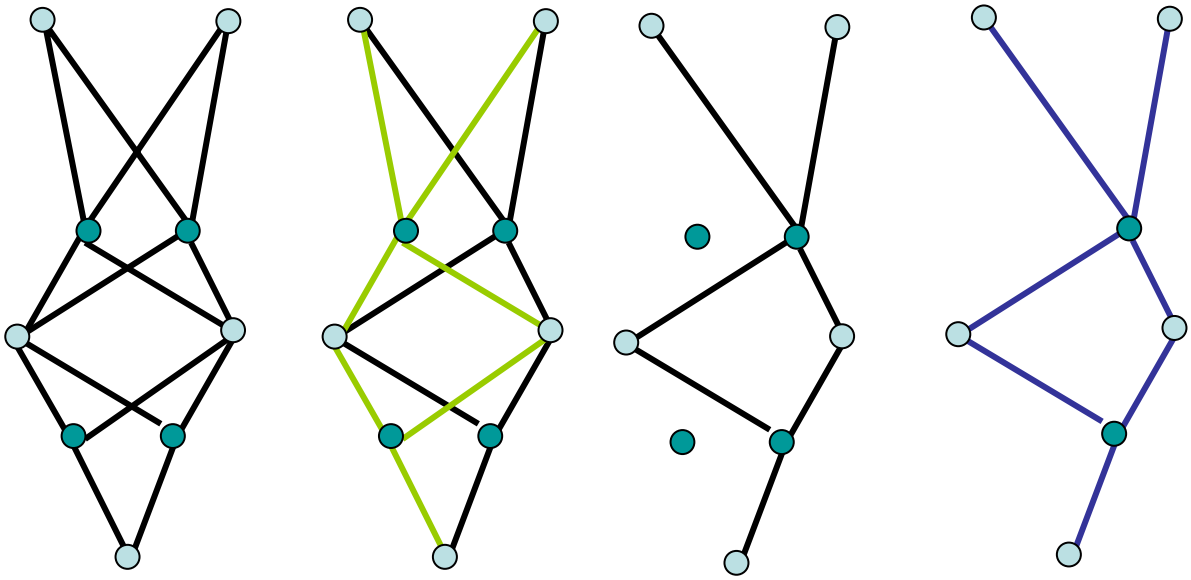

Abbildung 67 – Spanning Tree Algorithmus auf Modell

Die Abbildung Abbildung 67 verdeutlicht den Algorithmus. Auf der linken Seite ist das Ausgangsmodell wiederholt. Die beiden Webserver müssen dabei mit den beiden Datenbankservern verbunden sein, um mit diesen zu kommunizieren. Der Cluster ist hier vereinfacht. Die beiden Datenbankserver müssen mit dem Speichersubsystem verbunden werden, auf dem die Daten gespeichert sind.

Die zweite Grafik zeigt das Modell nach dem ersten Durchlauf des Algorithmus. Der erste Spannbaum ist grün eingefärbt. Dieser wird nun aus dem Modell entfernt. Dabei bleiben zwei Knoten erhalten, die nicht mehr mit den IT-Systemen verbunden sind. Dies ist in Grafik 3 dargestellt. Diese beiden Switche werden nun aus dem Modell entfernt. Es ergibt sich automatisch in der vierten Grafik ein zweiter Spannbaum. Wir haben so zwei voneinander unabhängige Netzwerkpfade beschrieben, die alle IT-Systeme des Modells unabhängig voneinander miteinander verbinden.

6.4 Integration anderer Modelle am Beispiel von ARIS EPKs

Wir haben in den vorigen Kapiteln bereits mehrmals ARIS EPKs als ein Beispiel für die Geschäftsprozessmodellierung beschrieben. Wir werden in diesem Abschnitt beschreiben, wie sich ARIS EPKs in unser Modell integrieren lassen.

Ereignisgesteuerte Prozessketten (EPKs) werden beschrieben durch einen bipartiten Graphen. Sie bestehen aus einer alternierenden Folge von Funktionen und Ereignissen. Ereignisse beschreiben dabei Vorbedingungen, die für Funktionen gelten. Die eigentliche Aktivität ist in der Funktion beschrieben. Darüber können EPKs logische Verknüpfungsoperatoren beinhalten, die verschiedene Möglichkeiten in einem Ablauf beschreiben. Zur Gliederung von EPKs stehen Unterprozesse zur Verfügung, mit denen EPKs entsprechend verfeinert werden können.

Die Integration von EPKs in unser Geschäftsprozessmodell ist sehr einfach möglich. Da EPKs als bipartite Graphen vorliegen, lassen sie sich direkt auf unser Geschäftsprozessmodell abbilden. Relevant sind für uns die Funktionen, da Ereignisse nur Vorbedingungen und keine Aktionen darstellen. Die Funktionen werden auf die einzelnen Geschäftsprozesse in unserem Geschäftsprozessmodell abgebildet. Die Abläufe durch die Ereignisse finden sich direkt in den

Abläufen unseres Geschäftsprozessmodells. Die Verfeinerungsbeziehungen in ARIS lassen sich direkt auf unsere Geschäftsprozesshierarchien abbilden. Die Tabelle verdeutlicht die Abbildung der verschiedenen Elemente der beiden Modelle aufeinander.

ARIS EPKs	Geschäftsprozessmodell
Ereignisse	Keine Entsprechung
Funktionen	Geschäftsprozesse
Verknüpfungen	Abläufe, die logischen Informationen der Verknüpfung werden nicht modelliert
Unterprozesse	Geschäftsprozesshierarchien

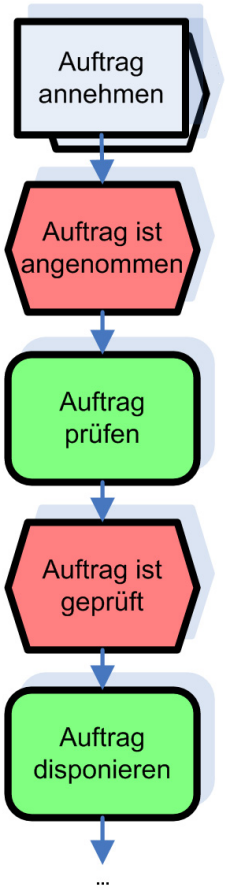

Abbildung 68 – ARIS EPK [Quelle; Wikipedia]

Damit lassen sich Abläufe und Hierarchien eines EPKs auf Elemente unseres Geschäftsprozessmodells abbilden. Darüber hinaus lassen sich alle Elemente unserer Geschäftsprozessmodelle auf ein EPK abbilden. Damit ist es auch möglich, EPKs direkt als ein Geschäftsprozessmodell in ein IT-Dienstmodell nach unseren Vorgaben zu integrieren und auf die Modellierung eines Geschäftsprozessmodells zu verzichten, wenn dieses als EPK bereits vorliegt. Wir geben in Abschnitt 9.3 ein Beispiel dafür an, wie ARIS EPKs direkt in unser IT-Dienstmodell zur Modellierung eines Geschäftsprozesses integriert werden können.

6.5 Weitere Anwendungen

Neben den hier beschriebenen Möglichkeiten kann das Modell auch an andere Anforderungen angepasst werden. Durch die beschriebene Möglichkeit der Integration von anderen Teilmodellen ist es zum Beispiel möglich, das Modell nur zur Modellierung von Beziehungen zwischen verschiedenen, speziellen Teilmodellen auf einer der Ebenen Infrastruktur, Software- und Geschäftsprozesse zu modellieren und durch das Funktionsmodell diese Abbildungen zu detaillieren. Dadurch wird eine Integration unterschiedlicher Modelle einer Ebene durch unser Modell ermöglicht, was mittelfristig zur Standardisierung der Modellierung beitragen kann.

Die beschrieben Funktionen zur Modellierung von Ausfällen sind eine Art der Modellierung von Verhalten im Modell. Diese lassen sich erweitern oder auch zur Simulation benutzen. Es sind aber auch zusätzliche Funktionen denkbar, die andere Aspekte modellieren.

Wir gehen im nächsten Kapitel auf einige Beispiele für die Modellierung mit dem Modell ein.

7 Beispiele der Anwendung des IT-Dienstmodells

Wir haben bei der Beschreibung unseres Modells bereits Beispiele für die Modellierung mit angegeben. Dieser Abschnitt illustriert anhand von weiteren Beispielen die Verwendung des Modells. Die Beispiele zeigen vor allem die Modellierung spezieller Aspekte und technische Besonderheiten einer IT-Infrastruktur.

7.1 Dynamische Softwarebeziehungen

Frameworks wie CORBA oder auch Webservices bieten mit einem CORBA Object Request Broker (ORB) oder UDDI einen Dienst, mit dem verteilte Komponenten in einer IT-Infrastruktur gefunden werden können. Die Dienste arbeiten als ein Verzeichnisdienst, der alle Softwarekomponenten mit ihren Schnittstellen und die Verteilung in der Infrastruktur kennt. Dazu werden die Softwarekomponenten in das Verzeichnis mit ihrer Adresse, unter der sie erreichbar sind und einer Schnittstellenbeschreibung eingetragen. Die Spezialität ist in diesem Fall, dass die Beziehung zwischen den Softwarekomponenten teilweise dynamisch ist. Client und Server finden sich über den ORB, dies kann zur Laufzeit erfolgen, die Beziehung wird nicht schon beim Deployment festgelegt.

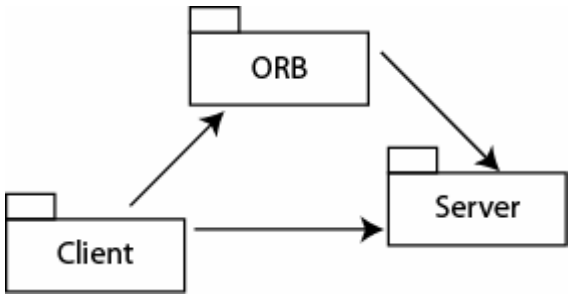

Abbildung 69 – Client/Server und ORB/UDDI

Benötigt ein System einen entsprechenden Dienst, so stellt es eine Anfrage an den ORB und erhält eine konkrete Instanz der Softwarekomponente zugewiesen, mit der das System dann interagiert. Abbildung 69 verdeutlicht diesen Vorgang. Die Interaktion mit der Serverkomponente erfolgt dann ohne einen Zugriff über den ORB[91]. Diese Zuweisung kann dynamisch sein, wenn mehrere Instanzen des Dienstes in der Infrastruktur existieren. Der ORB kann hier zum Beispiel Lastverteilungsmechanismen implementieren oder bei einem Ausfall einer Komponente die entsprechenden Anfragen auf funktionsfähige Instanzen umlegen.

Die Modellierung benötigt bei dieser Realisierung zwei Requirements. Neben der Modellierung des Requirements für die konkrete Instanz wird auch der ORB als Requirement modelliert. Die Modellierung der Beziehung zwischen Clientkomponente und ORB ist notwendig, da bei einem Ausfall des ORB auch kein neuer Zugriff auf die Serverdienste, auf die er den Zugriff ermöglicht, mehr möglich ist, da sie nicht mehr aufgefunden werden können. Für den ORB werden alle Schnittstellen für Komponenten, die im ORB registriert sind, als Requirement modelliert, um die Beziehung zwischen dem ORB und den Realisierungen zu modellieren.

Die Modellierung der konkreten Instanz bei den Softwarekomponenten ist notwendig, da sonst beim Deployment nicht geprüft werden kann, ob die Kanäle zu den jeweiligen konkreten Instanzen vorhanden sind. Eventuell ist der Zugriff auf den ORB möglich, aber eine Firewall verhindert den Zugriff auf die konkrete Instanz. In diesem Szenario muss das Deployment die entsprechende Anforderung erkennen, modelliert durch die Requirementbeziehung des ORBs zu den jeweiligen Softwarekomponenten.

[91] Die Implementierung, bei der alle Zugriffe weiterhin über den ORB abgewickelt würden, würde der Implementierung des Fassade Patterns entsprechen.

7.2 Installation Betriebsystem

Dieses Beispiel zeigt, wie die Informationen bei der Installation eines Betriebsystems auf einem Server innerhalb des Modells gepflegt werden. Unser Ausgangsszenario beschreibt eine Infrastruktur bestehend aus einem Intel x86 Server (Server 1) und einem Alpha AXP Server (Server 2). Beide sind direkt mit einem Patchkabel verbunden, das mit den Netzwerkkarten der Server verbunden ist. An diese Netzwerkkarten sind die Endpunkten 3 und 4 gebunden. Server 2 ist zusätzlich an eine externe Storageeinheit auf der Basis eine Fibrechannelnetzes mit 14 Festplatten und einer Kapazität von 2 Terabyte angeschlossen. Das Fibrechannelnetz wird durch eine direkte Verbindung auf der physikalischen Ebene modelliert und an diese die Endpunkte 1 und 2 gebunden.

Auf eine Modellierung der internen Struktur des Fibrechannelsystems wurde verzichtet. Die Server sind jeweils mit 2 Gigabyte Hauptspeicher ausgestattet, Server 1 besitzt eine 100 Gigabyte große Festplatte, Server 2 eine 10 Gigabyte Festplatte. Auf dem Server 1 soll Windows 2003 als Softwarekomponente ausgeführt werden. Die Abbildung 70 verdeutlicht das Beispiel.

Das Infrastrukturmodell wird durch folgende Funktionen erzeugt:

createITS(„Server1")

createITS(„Server2")

createITS(„FCStorage")

Formel 74 – Generierung von IT-Systemen

Dadurch werden die IT-Systeme im Infrastrukturmodell erzeugt. Im nächsten Schritt werden die Attribute der verschiedenen Systeme festgelegt.

Server1.setAttrib ("arch","x86")

Server1.setAttrib ("mem","2 GB")

Server1.setAttrib ("stor","100 GB")

Server2.setAttrib ("arch","axp")

Server2.setAttrib ("mem","2 GB")

Server2.setAttrib ("stor","10 GB")

FCStorage.setAttrib ("nrdisk","14")

FCStorage.setAttrib ("stor","2 TB")

Formel 75 – Festlegung der Attribute auf den IT-Systemen

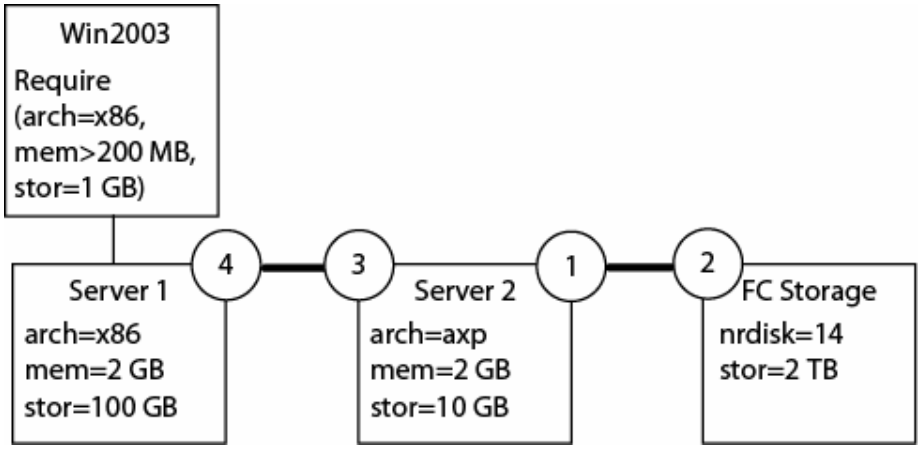

Abbildung 70 – Modell 2 Server, FC Storage, Win2003 Installation

Damit sind die Attribute der drei IT-Systeme definiert. Im nächsten Schritt definieren wir die Endpunkte und die Kanäle zwischen den drei IT-Systemen. Wir definieren eine sehr einfache Schnittstellenhierarchie, in der wir nur die Schnittstellenebenen Ethernet bzw. Fibrechannnel und für die Serververbindung eine IP Ebene definieren. Daraus ergibt sich die Modellierung der Endpunkte.

EP4= createEP([00:00:00:00:00:00]

EP3 = createEP([00:00:00:00:00:01]

EP1 = createEP([00:00:00:00:00:00:00:00:00:00]

EP2 = createEP([00:00:00:00:00:00:00:00:00:01]

BindeEndpunkt (EP4, Server1)

BindeEndpunkt (EP3, Server2)

BindeEndpunkt (EP1, Server2)

BindeEndpunkt (EP2, FCStorage)

BindeSchnittstelle (EP4, ([00:00:00:00:00:00] : 192.168.0.1))

BindeSchnittstelle (EP3, ([00:00:00:00:00:00] : 192.168.0.2))

Formel 76 – Definition der Endpunkte und Schnittstellen

An die Endpunkte können nun die Kanäle modelliert werden. Diese Kanäle beschreiben auf der untersten Hierarchieebene die physikalischen Verbindungen. Die logische Verbindung auf der TCP/IP Ebene zwischen den beiden Servern wird durch einen Kanal in der Kanalhierarchie modelliert.

K1 = CreateKanal (Server1[00:00:00:00:00:00], Server2 [00:00:00:00:00:01])

K2 = CreateKanal (Server2[00:00:00:00:00:00:00:00:00:00], FCStorage[00:00:00:00:00:00:00:00:00:01])

K3 = CreateKanal (Server1[00:00:00:00:00:00] : 192.168.0.1, Server2 [00:00:00:00:00:01] : 192.168.0.2)

addsubKanale (K1, K3)

Formel 77 – Definition der Kanäle

Damit sind die logischen und die physikalischen Verbindungen der IT-Infrastruktur mit ihren Schnittstellen modelliert. Im nächsten Schritt wird die sehr einfach modellierte Softwarekomponente Windows 2003 beschrieben.

Win2003 = createSK(„Windows 2003")

Win2003.setAttrib(„req","arch=x86")

Damit wird im Softwaremodell eine Anwendung Windows 2003 definiert. Darüber hinaus wird die Anforderung für eine Intel x86 Architektur definiert. Dies ist keine Vereinfachung, die Intel x64 Architektur wurde hier nicht weggelassen, sondern die Modellierung trägt der Tatsache Rechnung, dass die Unterstützung für andere Plattformen bei Windows auch von einer anderen Softwareversion realisiert wird. Die Modellierung als Softwarekomponente scheint widersprüchlich, da Windows eher einer Anwendung entspricht. Wir lassen jedoch die Zuweisung von Anwendungen an IT-Systeme nicht zu. Deswegen wird Windows hier als Softwarekomponente modelliert. Ist das Softwaremodell sehr feingranular und sollen auch einzelne Komponenten von Windows 2003, wie zum Beispiel das Active Directory oder der Internet Information Server modelliert werden, müsste in diesem Fall eine Softwarekomponente Windows Core eingefügt wer-

den, die die Basisfunktionen des Betriebsystems modelliert[92]. Im nächsten Schritt wird die Installation des Betriebsystems auf dem Server1 beschrieben. Dazu kann mit der Funktion requires geprüft werden, ob alle Requirements erfüllt sind

Requires(Server1, Win2003) = true

Requires(Server2, Win2003) = false

Formel 78 – Requirements prüfen

Der Server 2 hat eine Alpha AXP Architektur, deswegen kann Windows 2003 auf diesem Server nicht installiert werden. Die Anforderung und die Überprüfung durch die Funktion requires liefert hier einen Fehler, die Funktion connectSIT kann nicht ausgeführt werden.

connectSIT(Win2003, Server1)

Formel 79 – Deployment

Damit haben wir alle Schritte zur Erstellung eines Modells auf der Ebene der Softwarekomponenten und der IT-Infrastruktur detailliert vorgestellt. Im nächsten Abschnitt gehen wir auf einige spezielle Aspekte bei der Modellierung der IT-Infrastruktur ein.

7.3 Komplexes Serversystem

Das zweite Modell beschreibt ein komplexes Serverszenario. Das Modell, dargestellt in Abbildung 71, beschreibt ein System, bei dem zwei Server je zwei virtuelle Maschinen ausführen, die jeweils zu einem Cluster verbunden sind, um die Hochverfügbarkeit für die virtuellen Maschinen zu realisieren. Die virtuellen Maschinen greifen dabei auf ein hochverfügbares Speichersubsystem über redundante Pfade zu. Das Szenario beschreibt damit einen Scale-In Server und modelliert die Möglichkeit der Spiegelung von Ressourcen in Rechenzentren oder über zwei Rechenzentren. Die 4 Prozessoren der Hostsysteme sind im Verhältnis 3:1 auf die virtualisierten Instanzen aufgeteilt.

Der Server 1 hat als Attribute die Informationen über die verbauten Prozessoren und die Architektur des Systems. Darüber hinaus haben die physikalischen Server jeweils 2 Netzwerkkarten, bei denen eine zur Bereitstellung der Dienste und eine zur Anbindung an das Speichersubsystem verwendet werden.

Die Hierarchiebeziehung der beiden virtuellen Instanzen VSRV1 und VSRV2 zu Server 1 wird in der Grafik durch die Modellierung der Maschinen innerhalb des Servers 1 verdeutlicht. Die virtuellen Instanzen haben jeweils zwei virtuelle Netzwerkkarten, bei denen eine zur Anbindung an das Speichersystem dient und der zweite für die Bereitstellung der virtuellen Dienste. Die virtuellen Karten werden auf die jeweiligen physikalischen Netzwerkkarten abgebildet. Die Server VSRV1 und VSRV3 bzw. VSRV2 und VSRV4 sind jeweils zu einem Cluster verbunden. Dabei sind die virtuellen Prozessoren im Verhältnis 3:1 verteilt, sichtbar in der Modellierung durch die Attribute der virtuellen Maschinen. Dies sorgt dafür, dass die einzelnen physikalischen Server höher ausgelastet sind und im Falle eines Failovers vom aktiven System zum passiven Knoten die Last auf der physikalischen Maschine zu groß wird. Damit wird zwar im Falle eines Ausfalls die Leistung des ausgefallenen Dienstes geringer, dem steht aber die bessere Auslastung der physikalischen Server gegenüber, da die Leerlaufenden passiven Knoten des Clusters weniger Ressourcen blockieren. Die Cluster werden durch die virtuellen Endpunkte E16 bzw. E15 modelliert. An diese Endpunkte werden die Schnittstellen des hochverfügbaren Dienstes gebunden, die Endpunkte sind an die virtuellen Netzwerkkarten der virtuellen Maschinen gebunden, die die Dienste realisieren. Die Anbindung an das Speichersubsystem erfolgen über die zweiten virtuellen Netzwerkkarten der Server, bezeichnet durch die Endpunkte E6, E8, E12 und E14, die auf die physikalischen Endpunkte E4 bzw. E10 der jeweiligen physikalischen Server abgebildet werden.

[92] Windows 2008 Server führt genau diese Komponente dann auch als Rolle ein.

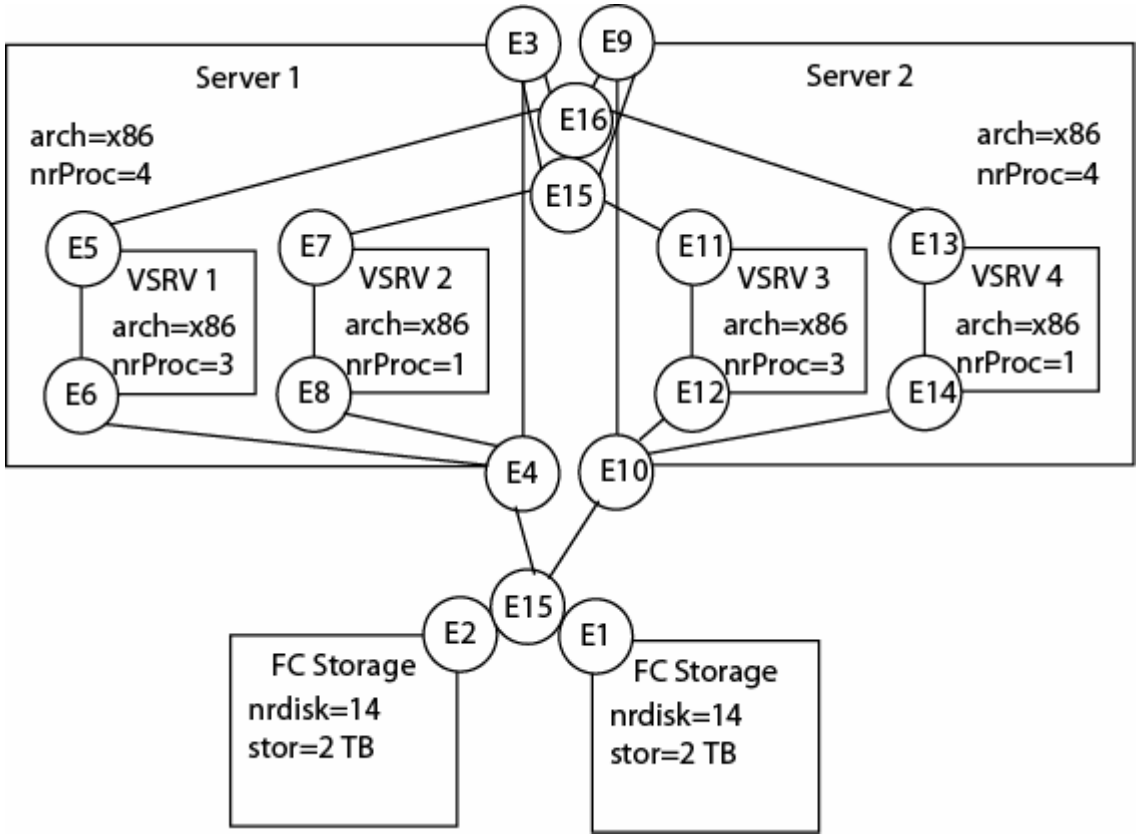

Abbildung 71 – Modell komplexer virtueller Serverinstanzen mit Clustering und HA-SAN

Das Speichersystem wird durch die beiden Storagesysteme realisiert. Beide haben jeweils eine physikalische Netzwerkverbindung, modelliert durch einen Endpunkt, E2 und E1. Diese beiden Endpunkte werden durch den Endpunkt E15 zu einem einzigen Speichersubystem zusammengefasst, der die Hochverfügbarkeit des Speichersubsystems modelliert. Die Eigenschaften der beiden physikalischen Speichersysteme werden in den jeweiligen Attributen beschrieben.

Eine alternative Modellierung wäre, ein virtuelles IT-System einzuführen, das aus den beiden Storagesystemen aufgebaut ist. Die Abbildung 72 verdeutlicht diese Modellierung. Das Speichersystem ist nun als ein IT-System aufgebaut, das die virtuelle Instanz des Speichersystems beschreibt. Die beiden physikalischen Endpunkte der Speichersysteme werden auch hier auf den Endpunkt E15 abgebildet, doch in diesem Fall ist dieser Endpunkt an ein virtuelles IT-System modelliert.

Die Frage, welche Art der Modellierung eingesetzt wird, hängt vor allem auch davon ab, wie die Hochverfügbarkeitslösung technisch realisiert wird, oder wie die Managementsysteme diese IT-Systeme modellieren. Findet die Spiegelung von den beiden Servern aus gesteuert statt, ist die erste Modellierung vorzuziehen, wird die Spiegelung von den beiden Speichersystemen transparent für die Server realisiert, ist das zweite Modell passender. Im Sinne unseres Modells sind beide Realisierungen identisch. Auf die Angabe der Formeln zur Erstellung des Modells haben wir verzichtet, diese ergeben sich analog zum vorigen Beispiel.

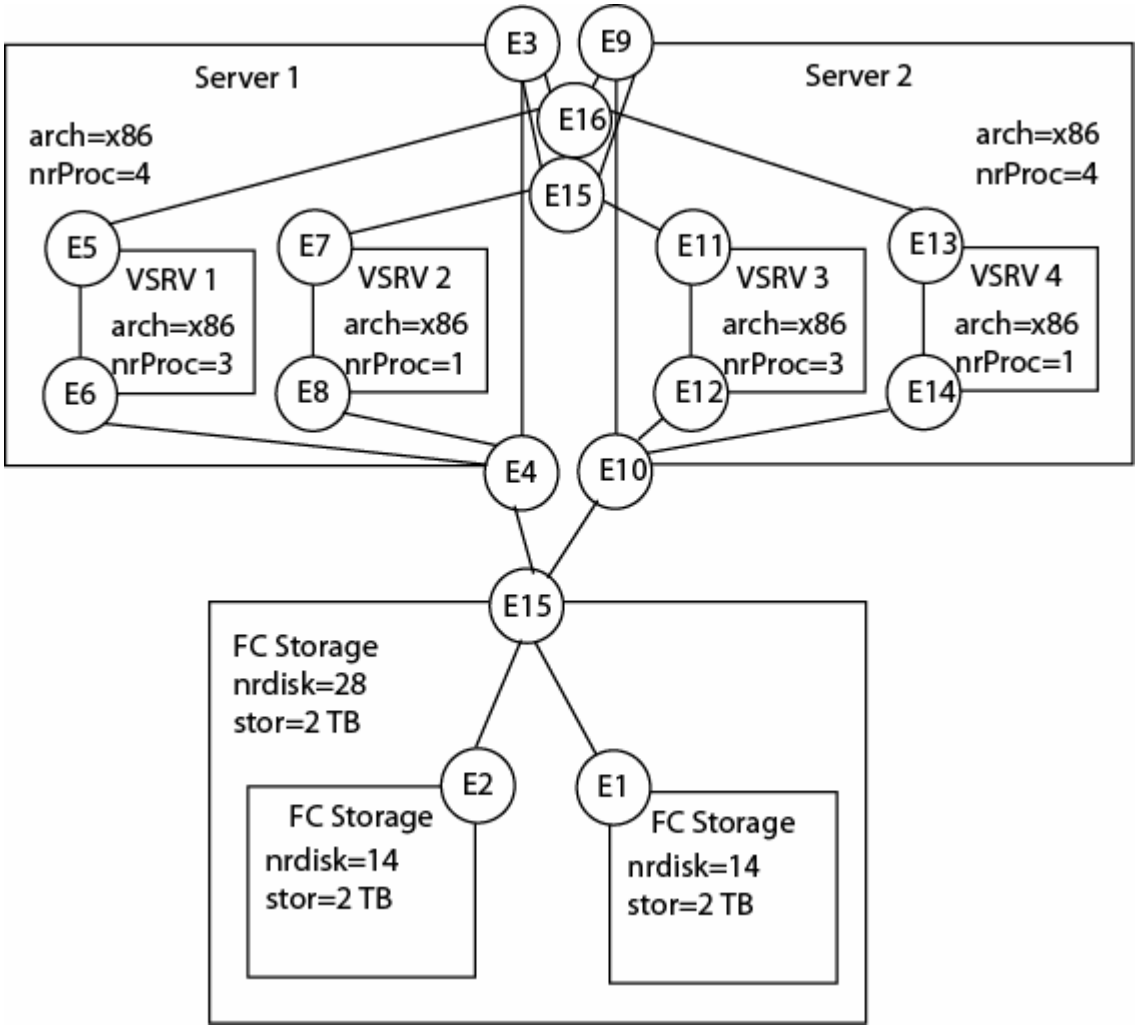

Abbildung 72 – Alternative HA-SAN Modellierung

7.4 Fazit

Diese Beispiele dienen vor allem als Einführung und sollen verdeutlichen, dass auch komplexe Infrastrukturen mit dem Modell beschrieben werden können. Wir werden im Abschnitt 9.2 noch weitere Beispiele einführen.

8 Vorgehensmodell

In diesem Kapitel stellen wir das Vorgehensmodell zur Erstellung eines Modells eines IT-Dienstes vor. Wie bereits in Kapitel 2.3.2 eingeführt, gehen wir von einem integrativen Entwicklungsprozess aus, bei dem Verantwortliche auf den einzelnen Ebenen aus ihren jeweiligen Teilmodellen ein Gesamtmodell eines IT-Dienstes entwickeln. Wir beschreiben kein Vorgehensmodell zur Geschäftsprozess- oder Softwareentwicklung, auf diesen Ebenen existieren bereits etablierte Verfahren, die dafür genutzt werden können, wie das Wasserfall- oder Spiralmodell in der Softwareentwicklung, ARIS oder allgemeiner Rahmenwerke wie das V-Modell XT. Gerade durch die Integration existierender Teilmodelle unterstützt unser Modell diese Ansätze, die ihre jeweiligen eigenen Modelle zur Verfügung stellen. Wir entwickeln kein Weltmodell für die Beschreibung eines IT-Dienstes, entsprechend ist auch das Vorgehensmodell auf die Modellierung der Beziehungen zwischen den Teilmodellen ausgerichtet und macht keine Vorgaben, wie die detaillierten Teilmodelle erstellt werden sollen.

Wir haben bereits beschrieben, dass es zwei grundlegend unterschiedliche Ausgangsszenarien bei der Erstellung eines Modells eines IT-Dienstes gibt.

1. Entwicklung oder Umsetzung neuer Dienste

   Werden neue Dienste entwickelt und findet die Entwicklung verschränkt mit unserem Modell statt, können Informationen aus der Entwicklung in unser Modell übernommen werden. Dies gilt sowohl für die Eigenentwicklung von Software, als auch für Projekte zur Einführung von Kaufsoftware in einem Unternehmen. Im Sinne unseres Modells sind beide Vorgehensweisen sehr ähnlich und wir werden beide mit einem Vorgehensmodell beschreiben. Besonderheiten bei den einzelnen Schritten werden wir entsprechend kennzeichnen. Eventuell kann es in diesem Szenario sogar vorkommen, dass sogar die IT-Infrastruktur völlig neu geplant wird. In den meisten Fällen werden jedoch bereits Teile der IT-Infrastruktur existieren, die entsprechend in das Gesamtmodell integriert werden müssen.

2. Beschreibung existierender Dienste in einer Infrastruktur

   Unser Modell kann auch eingesetzt werden, um eine existierende Infrastruktur zu beschreiben und das Management der Infrastruktur zu verbessern. In diesem Fall stehen oft keine Informationen aus der Entwicklung der verschiedenen Dienste zur Verfügung. Hier ist es wichtig, die existierenden Teilmodelle des IT-Dienstes, soweit diese vorhanden sind, zu vervollständigen und die Beziehungen zwischen den Teilmodellen deutlich zu machen. In vielen Fällen wird man hier auf detaillierte Informationen im Bereich des Infrastrukturmodells zurückgreifen können, eventuell sind auch die Informationen über die Softwareverteilung aus eingesetzten Managementsystemen verwendbar.

Wir haben bereits einen allgemeinen Prozess zur Gesamtmodellerstellung eines IT-Dienstes beschrieben. Die Abbildung 29 auf Seite 70 beschreibt, wie sich unser Modell in diesen allgemeinen, basierend auf ITIL spezifizierten Prozess integriert. Darauf aufbauend verdeutlicht die Abbildung 73, basierend auf der allgemeinen Einordnung einige der Erweiterungen unseres Modells. So werden die Elemente des Softwaremodells und des Geschäftsprozessmodells zuerst den Features des Funktionsmodells zugeordnet und basierend auf diesem gemeinsamen Wissen die Beziehungen zwischen den Ebenen beschrieben. Die Abbildung zwischen den Softwarekomponenten und der IT-Infrastruktur erfolgt dann wieder beim Deployment. Daraus entsteht ein Gesamtmodell eines IT-Dienstes, das die vier Ebenen beinhaltet und die entsprechenden Beziehungen beschreibt.

Wie auch beim allgemeinen Prozess ist hier eine Rückkopplung wichtig, da sich ein IT-Dienst über die Zeit an Veränderungen anpassen muss und diese Veränderungen möglicherweise auch alle Ebenen betreffen können.

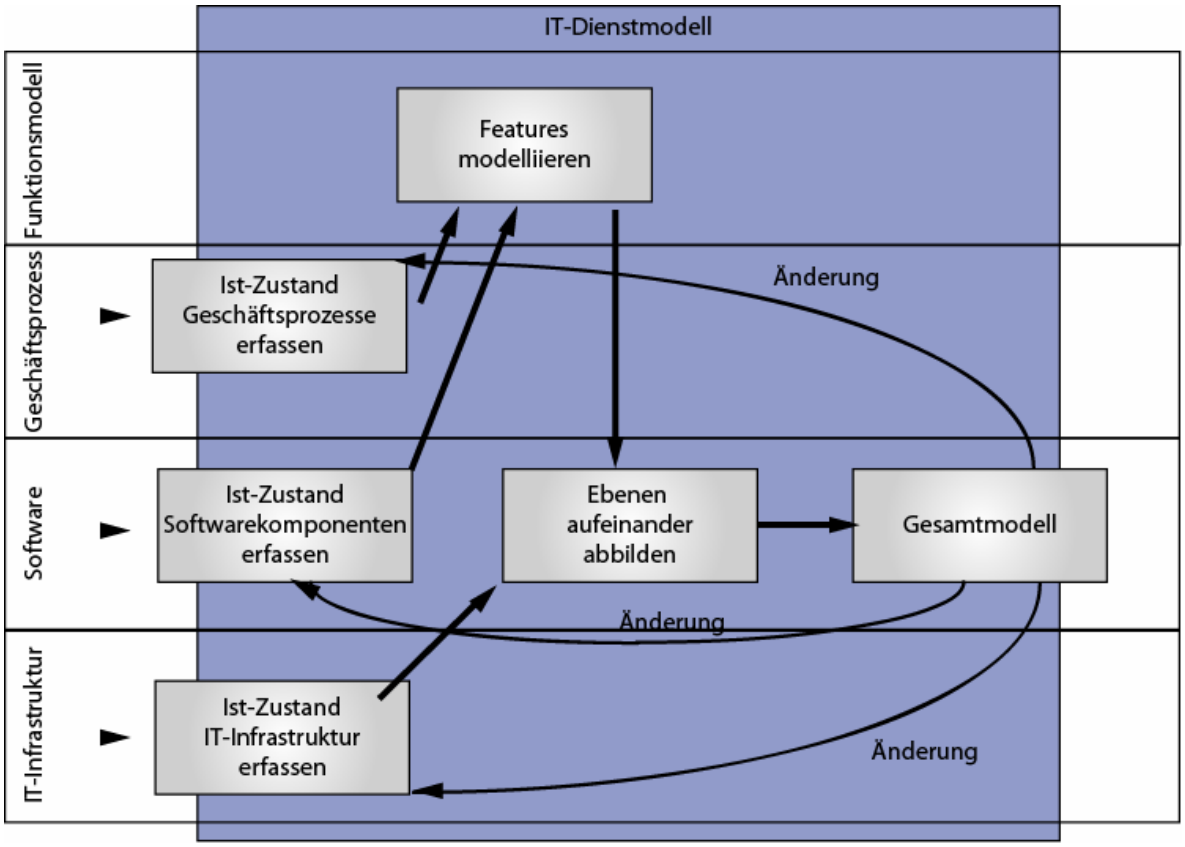

Abbildung 73 – Modell im allgemeinen Prozess

8.1 Schritte bei der Modellerstellung

Für beide Ausgangssituationen gilt, dass zuerst der Ist-Zustand der verschiedenen Modellebenen erfasst werden muss. Dabei werden sowohl die Objekte und Instanzen der verschiedenen Ebenen beschrieben, als auch deren Beziehungen zueinander analysiert.

Wie bereits in Kapitel 2.3.1 beschrieben, ist dies in ITIL und den dazugehörigen Tools die Aufgabe eines Application- oder Topology Scanners. Diese Erfassung muss auch auf den anderen Ebenen erfolgen, teilweise lässt sich das Softwaremodell aus den Informationen der IT-Managementtools übernehmen. Wenn für die Geschäftsprozesse bereits Modelle vorhanden sind, können auch diese als Ausgangbasis verwendet werden. Eventuell ist es aber auch notwendig, die Geschäftsprozesse erst zu modellieren, wenn die Steuerung und Betrachtung der IT-Dienste seither eher aus der technischen Perspektive erfolgt ist.

Anschließend werden die verschiedenen Ebenen aufeinander abgebildet. Es wird beschrieben, welche Softwarekomponenten auf welchen Server ausgeführt werden und welche Geschäftsprozesse durch diese realisiert werden. Darüber hinaus werden die Funktionen der Softwarekomponenten als Features im Funktionsmodell beschrieben und die beiden Teilmodelle aufeinander abgebildet. Häufig ist es hier sinnvoll, das Funktionsmodell in diesem Prozess zu erstellen, da eine Modellierung ohne Informationen über die tatsächliche Struktur meist schwieriger ist. Eventuell sind bei dieser Herangehensweise auch mehrere Durchgänge notwendig, bis das endgültige Modell so erstellt ist.

Wir werden in den nächsten beiden Abschnitten auf die jeweiligen Ausprägungen für die verschiedenen Ausgangsszenarien eingehen.

8.2 Das Modell im Softwareentwurfsprozess

Durch die Einbettung eines Gesamtmodells eines IT-Dienstes in den Entwicklungsprozess eines IT-Dienstes werden vor allem Medienbrüche vermieden, die bei der Weitergabe von Informationen immer wieder entstehen und durch die Informationen verloren gehen können.

Die Aufgaben des Softwareentwurfs, die für unser Modell am wichtigsten sind, sind

- Identifizierung und Analyse der zu unterstützenden Geschäftsprozesse

  In dieser Phase werden die durch den IT-Dienst zu unterstützenden Geschäftsprozesse festgelegt und beschrieben. Diese Aufgabe erfolgt durch den Geschäftsprozessverantwortlichen, der einen existierenden Geschäftsprozess anpasst oder einen neuen Geschäftsprozess entwickelt. Dabei sind vor allem die Struktur und die Abhängigkeit dieses Geschäftsprozesses von bereits existierenden Geschäftsprozessen wichtig. Die Modellierung dieser Geschäftsprozesse kann entweder mit einem speziellen Modell dieser Ebene oder unserem Geschäftsprozessmodell erfolgen.

- Beschreibung der Features der einzelnen Softwarekomponenten

  Nachdem entschieden ist, dass der Geschäftsprozess umgesetzt wird, muss die Abbildung in Software für die computergestützten Komponenten der Geschäftsprozesses erfolgen. Dazu werden die entsprechenden Teilprozesse des Geschäftsprozesses identifiziert und die Funktionen der jeweiligen Komponenten beschrieben. Die Funktionen werden als Features im Funktionsmodell festgehalten und die Beziehungen entsprechend modelliert.

  Zusammen mit dem Entwickler werden nun die Aufgaben der Softwarekomponenten festgelegt. Der Softwareentwickler entwirft die Architektur der zu implementierenden Anwendung und legt die Aufgaben der einzelnen Softwarekomponenten fest. Diese werden wieder auf die Features abgebildet, womit die Beziehung zwischen den Softwarekomponenten und den Geschäftsprozessen festgelegt ist.

  Wird keine neue Software entwickelt, sondern eine Kaufsoftware eingeführt oder adaptiert, werden hier die entsprechenden Softwarekomponenten der Kaufsoftware auf die Features abgebildet.

- Softwarekomponenten und die Gesamtarchitektur

  Vor der Implementierung wird die Gesamtarchitektur eines Softwaresystems, sowie seine interne Strukturierung festgelegt. Wichtig sind hier die verschiedenen Komponenten, die sich auf verschiedene Server verteilen lassen sowie die Anforderungen dieser Komponenten an ihre Laufzeitumgebung. Darüber hinaus sind für das Modell die angebotenen Schnittstellen, die bereits in der Infrastruktur ausgeführt werden, sowie die Basisdienste relevant, die die Softwarekomponenten benutzen. Zusammen mit den Abhängigkeiten der Softwarekomponenten untereinander werden die Informationen im Softwaremodell hinterlegt.

  Nach der Umsetzung in Source-Code steht das Anwendungssystem zum Deployment zur Verfügung.

- IT-Infrastruktur

  Eventuell existiert die Infrastruktur des IT-Dienstes noch nicht, in der dieser ausgeführt werden soll. Dann kann in diesem Zusammenhang jetzt das Infrastrukturmodell erstellt werden. Meist wird jedoch eine Infrastruktur bereits vorhanden sein, da IT-Dienste sehr selten „auf der grünen Wiese“ entwickelt werden. Diese IT-Infrastruktur muss im Infrastrukturmodell erfasst werden. Anschließend kann das Deployment im Modell durchgeführt werden. Damit können alle Anforderungen überprüft werden.

Nach diesen Schritten steht ein Modell eines IT-Dienstes zur Verfügung, das eventuell vorhandene oder entwickelte Modelle aus dem Softwareentwurf und der Geschäftsprozessmodellierung integriert und die Beziehungen eines IT-Dienstes von den Geschäftsprozessen über die Softwarekomponenten bis zur IT-Infrastruktur beschreibt.

8.3 Das Modell zur Unterstützung einer ITIL CMDB

Oft wird das Modell für einen IT-Dienst auch nachträglich entwickelt, Teilmodelle existieren dabei auf den Ebenen bereits. Hier ist es wichtig, diese Modelle entsprechend zu integrieren.

Wir werden im zweiten Schritt beschreiben, wie dies in einem Szenario erfolgen kann, in dem bereits eine ITIL konforme CMDB eingesetzt wird und ein Modell der Geschäftsprozesse existiert.

Wird eine ITIL-konforme CMDB eingesetzt, so steht sowohl ein Modell der IT-Infrastruktur zur Verfügung, als auch ein Modell, das die Verteilung der Softwarekomponenten in dieser Infrastruktur beschreibt. Aus diesem lässt sich sehr einfach unser IT-Infrastrukturmodell ableiten. Auch ein Teil des Softwaremodells lässt sich so erstellen. Es fehlen in diesem Fall im Softwaremodell aber meist die Informationen über die Abhängigkeiten und die Beziehungen zu den Geschäftsprozessen. Diese Informationen müssen in diesem Bottom-Up Prozess erarbeitet werden.

Wichtig sind in diesem Zusammenhang drei Beziehungen:

- Die Abhängigkeiten der Softwarekomponenten zueinander im Sinne von Hierarchien wie Containerbeziehungen zu einer Laufzeitumgebung.

  Diese Informationen lassen sich teilweise aus den Managementsystemen der jeweiligen Basiskomponenten ableiten.

- Die Abhängigkeiten der Softwarekomponenten voneinander im Sinne der Aufrufbeziehungen und der Abläufe der Softwarekomponenten.

  Diese Informationen lassen sich teilweise aus dem Managementsystemen der Basiskomponenten ableiten, so besitzen Anwendungen, die sich auf eine Middleware wie CORBA abstützen hier entsprechende Client-Stubs, die Informationen über die Beziehungen enthalten. Diese lassen sich hier benutzen, ebenso wie Infrastrukturkomponenten wie Verzeichnisdienste, Object Request Broker (ORB) oder Logbücher von Webservern bei Webservice-basierten Systemen.

  Hier entwickelt sich ein neues Forschungsgebiet, das weitere Ansätze entwickelt, falls überhaupt keine Informationen über die Software mehr zur Verfügung steht.

- Beziehungen zwischen den Softwarekomponenten und den Geschäftsprozessen.

  Gerade bei feingranularen Softwarekomponenten ist hier die Dokumentation der Software oder das Wissen des Entwicklers sehr wichtig. Ansonsten können die Beziehungen zwischen den Ebenen nur sehr grobgranular modelliert werden.

All diese Informationen müssen für das Softwaremodell gesammelt und beschrieben werden. Häufig müssen hier alle Verantwortlichen in einem Reegineeringprozess die entsprechenden Informationen erarbeiten. Die Abbildung auf das Geschäftsprozessmodell führt in diesem Ansatz zu einem vollständigen Modell eines IT-Dienstes.

8.4 Ergebnis

Wir haben in den vorangegangenen Kapiteln unser Modell erarbeitet und beschrieben. Darüber hinaus haben wir hier zwei Vorgehensmodelle beschrieben, die die Modellierung der Beziehungen zwischen den verschiedenen Teilmodellen eines IT-Dienstes erlauben. Dabei geht es uns vor allem um die Modellierung der Beziehungen, wir geben kein detailliertes Vorgehensmodell für die einzelnen Ebenen an.

Im nächsten Kapitel gehen wir auf die Evaluiierung unseres Modells ein und bewerten unser Modell anhand der aufgestellten Anforderungen.

9 Evaluierung des Modells

Wir haben in den vorangegangenen Abschnitten unser Modell vorgestellt und Beispiele für die Verwendung angegeben. In diesem Kapitel gehen wir nochmals auf die von uns aufgestellten Anforderungen für ein Modell zur Modellierung von IT-Diensten ein und bewerten unser Modell anhand dieser Anforderungen.

Darauf aufbauend beschreiben wir Prototypen, die zur Verdeutlichung spezieller Aspekte des Modells implementiert wurden. Ein größeres Beispiel zur Evaluierung des Modells und als Nachweis für die Einsetzbarkeit wurde im Rahmen einer Diplomarbeit zusammen mit Siemens Business Services (SBS) eine konkreter Geschäftsfall von SBS modelliert. Wir stellen in diesem Kapitel eine Zusammenfassung der Arbeit vor und analysieren die Ergebnisse dieser Diplomarbeit.

Neben den Prototypen, die vor allem der Verdeutlichung von Teilaspekten und Konzepten des Modells dienen und deren Realisierbarkeit beweisen, wurde das Modell als Basis für ein Tool des Lehrstuhls Brügge verwendet. Wir stellen hier kurz die Ergebnisse vor.

9.1 Die Anforderungen

Wir haben in Kapitel 3 die Anforderungen an unser Modell definiert. Wir wollen unser Modell nun mit diesen Anforderungen bewerten.

Die aufgestellten fachlichen Anforderungen waren:

1. Modellierung von Geschäftsprozessen und ihrer gegenseitigen Abhängigkeiten, um Zusammenhänge zwischen den Geschäftsprozessen zu beschreiben.

   - Durch unser Geschäftsprozessmodell ist die Modellierung von Geschäftsprozessen, ihren Verschachtelungen und Abläufen möglich.

2. Modellierung der Struktur und der möglichen Abläufe von Softwaresystemen und damit Beschreibung der Beziehungen von Softwarekomponenten. Modellierung von Containerbeziehungen, um Softwarekomponenten, die in einer Laufzeitumgebung ausgeführt werden, modellieren zu können.

   - Durch das Softwaremodell werden die Softwarekomponenten modelliert, die Abläufe befinden sich in den modellierten Aufrufbeziehungen der Softwarekomponenten untereinander. Die Containerbeziehungen finden sich in den Hierarchien der Softwarekomponenten im Softwaremodell.

3. Beschreibung der IT-Infrastruktur mit der physikalischen und der logischen Netzwerkinfrastruktur, um Fehler, die verschiedene Komponenten betreffen, analysieren zu können. Dazu gehört auch die Beschreibung von möglichen Abläufen und Firewallsystemen, im Gegensatz zur grobranularen Modellierung wie in ARIS.

   - Das Infrastrukturmodell erlaubt die Modellierung von IT-Systemen und aktiven Netzwerkkomponenten. Durch die Möglichkeit der Verschachtelung ist auch die Möglichkeit der Modellierung interner Strukturen von Serversystemen oder von virtuellen Infrastrukturen möglich.
   - Die Feingranulare Modellierung der Kanäle und der Kanalhierarchien ermöglicht auch die Modellierung von partitionierten Netzwerken. Darüber hinaus werden hier auch die virtuellen Netze von Virtualisierungslösungen beschrieben und auf die physikalische Infrastruktur abgebildet. Ausfälle lassen sich so auch für virtuelle Systeme modellieren.
   - Wie in Abschnitt 6.1 beschrieben, lassen sich auch Firewallregeln in unser Modell integrieren.

4. Verbindung der Ebenen eines IT-Dienstes durch ein Modell, das die Ebenen Geschäftsprozess, Softwarekomponenten und IT-Infrastruktur verbindet und damit eine Modellierung aller Ebenen, die zu einem IT-Dienst beitragen.

- Die Abbildung der Beziehungen zwischen den Ebenen ist eines der Hauptmerkmale unseres Modells. Die entsprechenden Funktionen sind bei den verschiedenen Teilmodellen angegeben.

5. Integration von existierenden, auf den jeweiligen Ebenen vorhandenen Modellen zur Modellierung der jeweiligen Ebene in die Modellierung der Beziehungen und Abhängigkeiten.

- Wir haben in Kapitel 6.4 beschrieben, wie sich ARIS Modelle auf unser Modell abbilden lassen. Die Namensschemata von CIM lassen sich direkt in die Attribute der Teilmodelle integrieren, bzw. ein CIM Modell auf das Infrastrukturmodell abbilden und so die Informationen eine ITIL CMDB integrieren.

6. Modellierung und Beschreibung der Auswirkung von Ausfällen, um nicht nur die Abhängigkeiten der Komponenten voneinander, sondern auch die Funktion von Komponenten zu modellieren.

- Unsere Fail() Funktionen ermöglichen, je nach Implementierung in einem Tool, eine sehr mächtige Modellierung der Auswirkung von Ausfällen.

7. Erkennung von unterschiedlichen Implementierungen derselben Funktionalität, um einen vorhandenen Bedarf zur Standardisierung zu erkennen. Beschreibung von Standards zur Vorgabe von Pattern bei der Planung neuer IT-Systeme.

- Das Funktionsmodell abstrahiert die Funktionen der verschiedenen Implementierungen. Durch eine Überprüfung, ob ein Feature auf mehrere unterschiedliche Implementierungen auf der Ebene der Software- und der IT-Infrastruktur abgebildet wird, lässt sich erkennen, ob verschiedene Systeme für die Realisierung derselben Funktion eingesetzt werden.

Unser Modell erfüllt damit alle fachlichen Anforderungen, die wir definiert haben, um ein Modell zur Abbildung der Beziehungen zwischen den verschiedenen Ebenen eines IT-Dienstes zu realisieren. Wir betrachten nun die Anforderungen an die Nutzung, die wir für unser Modell aufgestellt haben.

Die Anforderungen sind:

1. Intuitiv verständliches Modell

- Die Verwendung der Graphentheorie als die Basis des Modells erleichtert auch für ungeübte Anwender den Einstieg. Die Netzwerkinfrastruktur und die Hierarchien dieser Struktur, die sich aus unterschiedlichen, teilweise nicht leicht verständlichen Unterschieden in der technischen Realisierung ergeben, werden in den Schnittstellenhierarchien versteckt und so leichter verständlich. Damit wird die Handhabung deutlich vereinfacht, ohne die Mächtigkeit einzuschränken.

- Das Modell sorgt dafür, dass die technischen Rahmenbedingungen automatisch eingehalten werden und kein tiefgreifendes Verständnis erforderlich ist.

2. Einfache grafische Darstellung

- Die Verwendung der Graphentheorie ermöglicht eine direkt grafische Darstellung.

3. Integration in ein Tool zum IT-Management

- Unser Modell lässt sich direkt in ein Tool zum IT-Management einsetzten, wir werden im Abschnitt 9.4 noch ein entsprechendes Beispiel vorstellen. Darüber hinaus lässt sich unser Infrastrukturmodell direkt auf existierende Schemata, wie den in CMDBs weitverbreiteten CIM-Standard abbilden.

4. Konsistenzsicherung durch das Modell

- Die Funktionen zur Erstellung und Veränderung unseres Modells stellen von sich aus die Konsistenz des Gesamtmodells sicher.

Unser Modell erfüllt damit sowohl die fachlichen Anforderungen, als auch die Anforderungen an die Nutzbarkeit für ein Modell zur Modellierung von IT-Diensten. Wir werden im nächsten Abschnitt auf einige Prototypen eingehen, die belegen, dass das Modell und die Modellideen auch in Tools entsprechend umgesetzt werden können.

9.2 Prototypen

Um die Realisierbarkeit des Modells zu belegen, wurden im Rahmen mehrerer Systementwicklungsprojekte Prototypen implementiert, die Teilaspekte und Ideen des Modell realisieren. Diese Prototypen sind dabei voneinander losgelöst. Sie könnten als Basiskomponenten für das in der Einleitung beschriebene Tool zur Verwaltung von IT-Infrastrukturen dienen. Im Rahmen der Arbeit wurden sie vor allem benutzt, um die Realisierbarkeit einzelner Aspekte zu belegen.

9.2.1 Regelgenerierung ISA Server

Die Ausgangssituation dieses Prototyps ist die Bereitstellung von webbasierten IT-Diensten am Lehrstuhl Prof. Broy. Dabei werden von Studenten und Mitarbeitern .NET basierte Webapplikationen entwickelt, die dann von der Systemverwaltung betrieben werden. Dieses Modell entspricht hier dem Hosted-Service, wie er auch oft in Unternehmensumgebungen oder in Outsourcing-Beziehungen vorgefunden wird.

Um diese Webanwendungen sicher zur Verfügung zu stellen, ist eine entsprechende Serverinfrastruktur notwendig, die durch Sicherheitsmechanismen wie Firewalls die Dienste vor unberechtigtem Zugriff schützt. Dazu wird in diesem Beispiel der Microsoft ISA Server 2004 eingesetzt.

Viele Sicherheitslücken, die in den letzten Jahren aufgetreten sind, basierten auf überlangen URLs, die an den Server geschickt wurden und die zu Buffer Overflows im Webserver geführt haben. Andere Probleme haben sich vor allem aus Komponenten ergeben, die auf dem Server fälschlicherweise aktiviert waren und so den Zugriff auf das System ermöglicht haben.

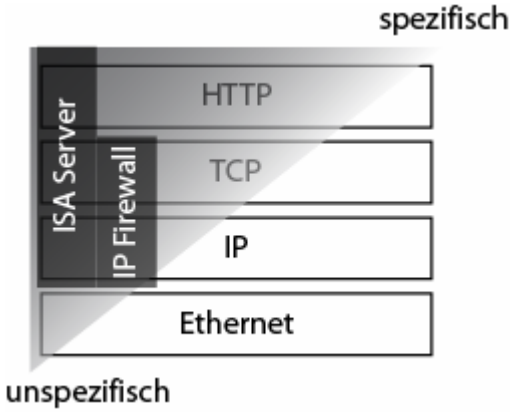

Abbildung 74 – Layer 3 / 7 Firewall

Der ISA Server ermöglicht hier einen sehr feingranularen Schutz des Systems. Die Firewall, die sowohl die Funktionalität einer Layer 3 Firewall als auch einer Layer 7 Firewall[93] bietet, kann einzelne Webseiten, Bilder und andere Inhalte schützen. Der Administrator kann dazu den Zugriff auf Inhalte eines Webservers sehr feingranular freischalten, indem er jede Seite einzeln sichern kann. Darüber hinaus können auch einzelne Befehle des http-Protocols deaktiviert werden, bzw. Fingerabdrücke von bekannten Angriffen dazu benutzt werden, diese Zugriffe bereits an der Firewall zu sperren. So können Angriffe, die auf fehlerhaft oder fälschlicherweise aktivierte Dienste zugreifen, wirksam verhindert werden. Darüber hinaus werden ungültige URLs gefiltert und nicht mehr an die tatsächlichen Webserver weitergeleitet. Es sind damit nur noch

[93] Diese Funktion wird manchmal auch als Application Gateway bezeichnet.

die spezifizierten URLs gültig, diese werden von der Firewall an den Webserver weitergeleitet, alle anderen Anfragen werden ausgefiltert.

Schwierig ist hier für den Administrator festzustellen, welche Inhalte er publizieren muss. Darüber hinaus enthalten die Anwendungen möglicherweise auch Webservices oder andere Inhalte, die nur intern verwendet werden und deshalb an der Firewall überhaupt nicht freigegeben werden sollen.

Der hier vorgestellte Prototyp dient dazu, die Wiederverwendung von Informationen während der Entwicklung eines Dienstes beim Betrieb der Software zu beschreiben. Webanwendungen sind hier als Beispiel sehr gut geeignet, das zum einen die Zahl der Informationen, die aus dem Code geparst werden müssen, eingeschränkt ist und zum anderen das Wissen und die Funktion auch auf Anwendung übertragen werden kann, die mit Webservices realisiert werden.

Verteilte Anwendungen verwenden so genannte Stubs, die eine lokale Realisierung einer entfernten Schnittstelle enthält. Die lokale Anwendung interagiert dabei mit dem Stub, der die Kommunikation mit dem Server abwickelt. Damit wird der Entwickler von der Implementierung der Netzwerkfunktionen entlastet, diese Funktionen stellt ein entsprechendes Framework wie CORBA oder die .NET Webservices zur Verfügung. Genau diese Stub enthalten nun die Informationen über die Verteilung, die dieser Prototyp auswerten kann.

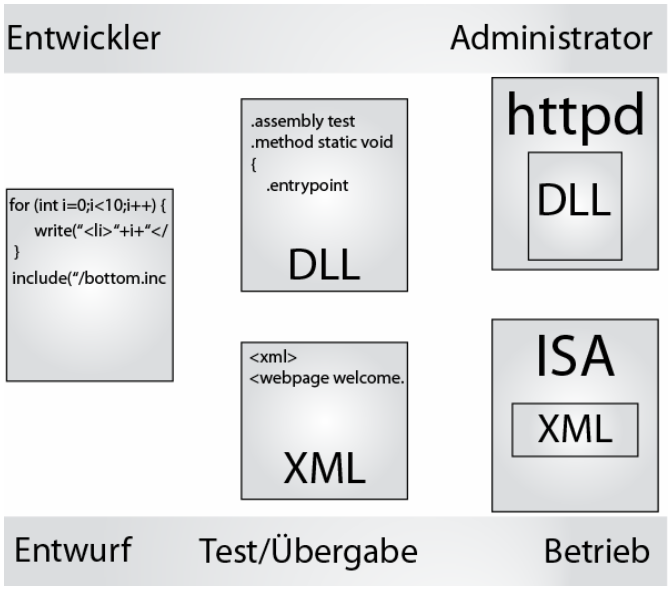

Abbildung 75 – Szenario Prototyp

Das als Prototyp implementierte Werkzeug ermöglicht hier die Weitergabe der Metainformationen über die Applikation. Der Entwickler wird damit in die Lage versetzt, den von ihm entwickelten Source Code zu parsen[94]. Daraus entsteht eine XML Datei, die alle Dateien der Webanwendung enthält. Damit lassen sich die Dateien beschreiben, die der Administrator an der Firewall freischalten muss. Darüber hinaus werden die verschiedenen Applikationen und Services erkannt, die die Anwendung bereitstellt oder benutzt. Jede Applikation und jeder Webservice werden für sich als Bündel zusammengefasst. Dies entspricht später auch dem Deployment auf dem Webserver, auf dem diese Bündel die kleinste mögliche Einheit des Deployments beschreiben.

Der Entwickler kann diese Bündel mit verschiedenen Standardvorgaben hinterlegen. Er kann so festlegen, welche Bündel der Administrator freigeben soll und welche nur zur internen Verwendung bestimmt sind und an der Firewall blockiert werden müssen. Neben dem Code der An-

[94] Die genau technische Beschreibung des Parsers findet sich in [Pag06].

wendung gibt der Entwickler auch die abstrakte Beschreibung in Form einer XML Datei weiter, die die Metainformationen enthält.

Der Administrator kann nun die verschiedenen Anwendungen und Services auf die Server verteilen. Mit dem zweiten Teil des Prototypen, der XML Beschreibung der Abhängigkeiten untereinander und dem Wissen über die Verteilung in der konkreten Infrastruktur kann er die richtigen Firewalleinstellungen für den ISA Server generieren. Dazu werden den Bündeln, die vom Entwickler in der XML Datei hinterlegt sind, die IP Adressen zugewiesen, wie diese in der Infrastruktur installiert wurden. Zusammen mit den Informationen, welche Seiten jeweils an der Firewall freigeschalten werden müssen, entsteht als Ergebnis für den Administrator ein Satz Firewallregeln, die er direkt in den ISA Server übernehmen kann.

Mit dem Tool ist damit eine Weitergabe von Handlungsanweisungen für den Administrator möglich, die weit über die Weitergabe in Form einer Textdatei hinausgehen. Die Informationen werden in Form eines Modells weitergegeben, das mit Tools für die Verwaltung direkt mit den Managementtools der IT-Infrastruktur und der Anwendungsbasissoftware interagiert. Das Szenario lässt sich direkt auf Webservice-basierte Anwendung ausweiten, die mit .NET realisiert wurden.

Der Prototyp belegt, dass sich sogar aus dem Source-Code einer Anwendung Informationen entnehmen lassen, die in ein Modell eines IT-Dienstes integriert werden können. Die XML-Datei des Prototypen stellt damit ein unserem Softwaremodell sehr ähnliches Modell dar, es ist aber an das spezielle Szenario angepasst und enthält weniger Informationen als unser Softwaremodell. Die prinzipielle Möglichkeit der Integration dieser Informationen in ein Modell zur Beschreibung der Abläufe und die Wiederverwendung dieses Modells beim Betrieb der Dienste ist aber durch den Prototypen belegt.

9.2.2 IT-Modeller

Der IT-Modeller ist eine prototypische Implementierung eines grafischen Modellierungstools für das IT-Dienstmodell mit den DSL Tools von Visual Studio 2005. Damit ist es möglich, das vorgestellte Modell toolgestützt zu implementieren. Ein grafischer Designer bietet dabei die Möglichkeit, die einzelnen Teilmodelle sowie die Beziehungen zwischen diesen zu modellieren.

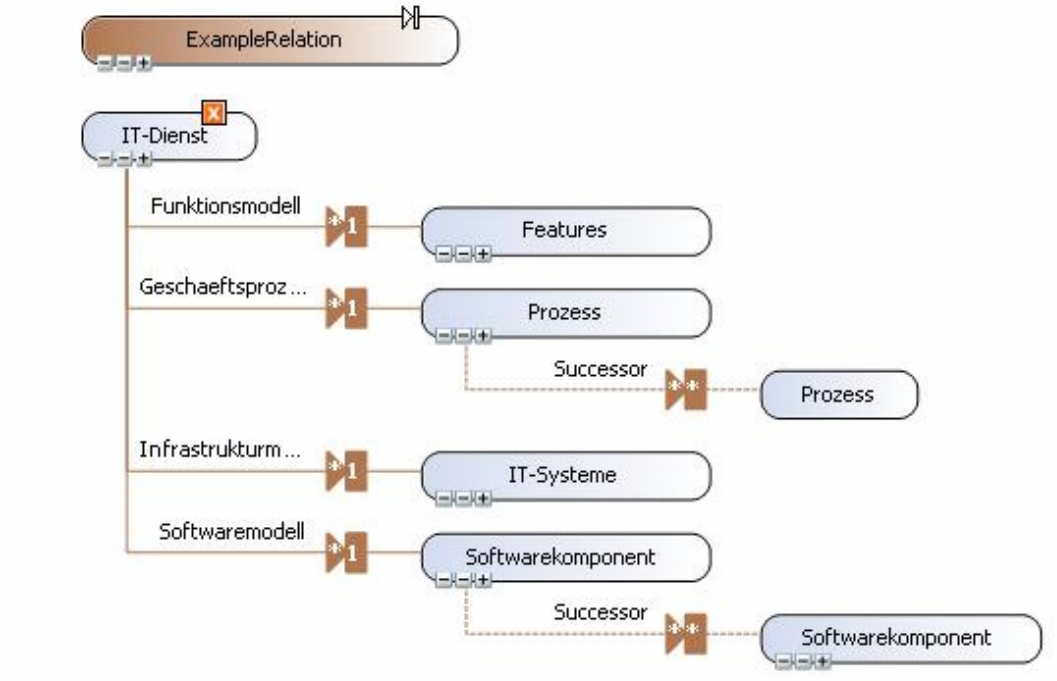

Abbildung 76 – DSL Definition IT-Modeller

Die Visual Studio DSL Tools sind ein Framework zur Realisierung von domänenspezifischen Sprachen (DSL's). Unser Modell wird in diesem Prototypen genau in eine solche DSL überführt. Unsere vorgestellten Funktionen werden dabei innerhalb des Designers hinterlegt. Alle Teilmodelle können wie in den vorigen Kapiteln vorgestellt implementiert werden.

Gerade die grafische Modellierung kann den Einstieg in ein Modell deutlich vereinfachen, da werkzeugunterstützt und grafisch direkt am Bildschirm modelliert werden kann. Der Benutzer kann die Komponenten der einzelnen Teilmodelle auf die Arbeitsfläche ziehen, ihre Attribute bearbeiten und die Abhängigkeiten zwischen den Komponenten durch das Zeichnen von Linien modellieren. Neben der physikalischen IT-Infrastruktur und den Kanälen erfolgt auch die Modellierung der Beziehungen zwischen den Teilmodellen auf diesem Weg. Gleiches gilt für die anderen Teilmodelle.

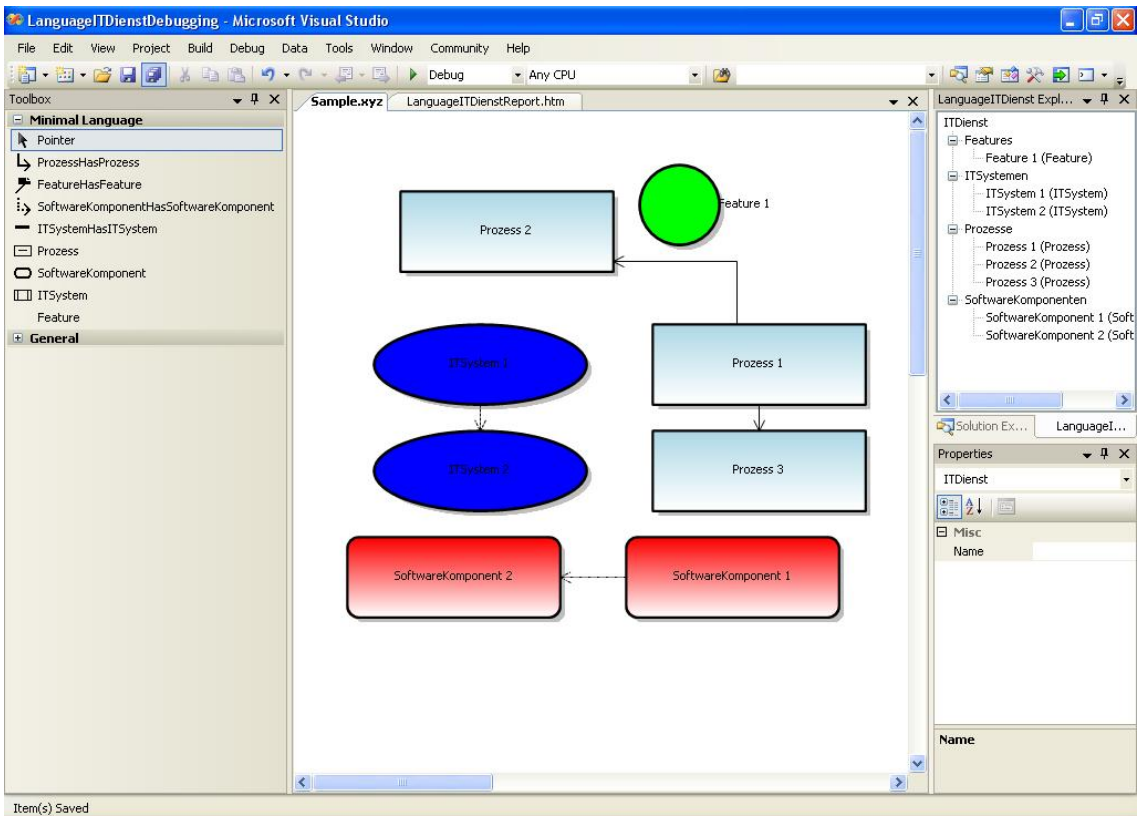

Abbildung 77 – IT-Modeller für den Anwender

Im Hintergrund führt der IT-Modeller die in den vorigen Kapiteln vorgestellten Funktionen aus und stellt so die Konsistenz des Modells sicher. Ist das Gesamtmodell eines IT-Dienstes beschrieben, gibt der IT-Modeller dieses Modell als eine XML Datei aus, die in anderen Tools entsprechend weiterverwendet werden kann.

Der IT-Modeller ist damit eine erste Stufe eines grafischen Modellierungswerkzeuges, das die Realisierbarkeit und die Nutzbarkeit des Modells in einem entsprechenden Tool belegt. Er kann als Ausgangspunkt für andere Tools verwendet und so in eine entsprechende Werkzeugkette integriert werden. Durch die Verwendung der DSL Tools und einer XML Datei als Codebasis ist die Weitergaben von Informationen einfach möglich. Die XML Datei lässt sich anschließen auch wieder in den grafischen Editor importieren. So können Veränderungen, die extern erfolgt sind, wieder direkt als Modell angezeigt werden.

9.2.3 Virtual Server Steuerung

Wir haben bereits beschrieben, dass virtuelle Infrastrukturen einen deutlich höheren Aufwand an Managementsysteme stellen, da auf einem Server viele virtuelle Instanzen ausgeführt werden. Diese Instanzen müssen entsprechend bereitgestellt werden und ihr Betrieb überwacht werden.

Am Lehrstuhl wird ein Tool entwickelt, das die Steuerung und die Bereitstellung von virtuellen Maschinen auf einem Virtual Server 2005 R2 Cluster ermöglicht. Neben einer automatisierten Erstellung der virtuellen Maschinen im iSCSI basierten SAN werden hier auch Funktionen implementiert, um die Hostsysteme und die einzelnen Server im Falle von Security Updates neu starten zu können, ohne die darauf aufbauenenden Services zu beeinflussen. Dazu werden Clustersysteme und Load-Balance Konfigurationen eingesetzt, bei denen jeder Dienst von mindestens einem Server ausgeführt wird.

Aufbauen auf einem Modell, das mit dem IT-Modeller gepflegt wird, erledigt das Managementsystem diese Aufgaben.

Der Prototyp belegt, dass unser Modell auch als Basis eines entsprechenden Managementwerkzeugs eingesetzt werden kann. Erweiterungen zur Verbesserung der Auslastung der einzelnen Server oder zur Rekonfiguration bei veränderten Lastprofilen sollen in Zukunft in das Tool integriert werden.

9.3 Evaluierung des Modells am Beispiel SBS P@ris

Neben den beschriebenen Prototypen wurde das Modell auch in einem größeren Rahmen evaluiert. Im Rahmen der Diplomarbeit „Analyse und Modellierung von Geschäftsprozessen im Rahmen des Projektes SBS SAP Konsolidierung“ [Kunze06] wurde das Modell zur Modellierung eines Geschäftsprozesses der Siemens Business Services (SBS) verwendet. Die Arbeit wurde im Rahmen der TUM/SBS Kooperation „Dynamic Value Webs for IT-Services“, einem Projekt der CKI [CKI] Initiative der TU München und der Siemens AG erstellt.

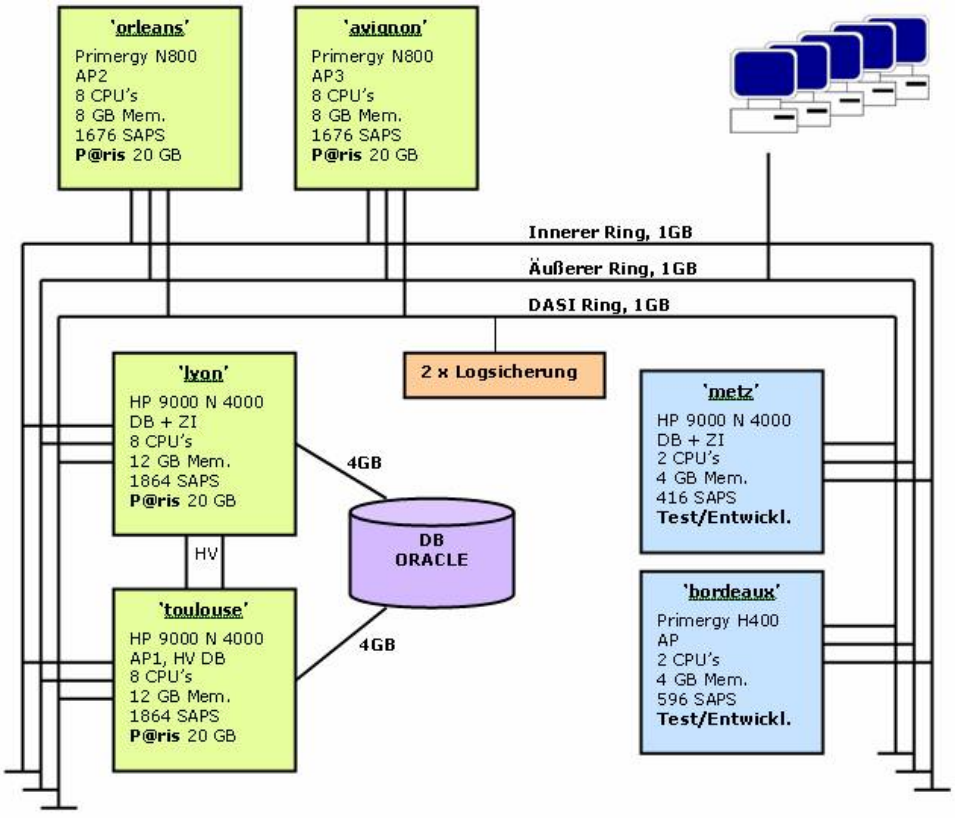

Abbildung 78 – P@RIS Infrastruktur [Kunze06]

Innerhalb der Arbeit wurde P@RIS evaluiert. P@RIS ist ein SAP R/3 System, das die Abbrechung des Betreibergeschäfts der SBS realisiert. Innerhalb des Systems werden alle erbrachten IT-Dienstleistungen erfasst und über die jeweiligen Kostenstellen intern verrechnet. Dazu werden Daten aus verschiedenen Vorsystemen in P@RIS übernommen und dort weiterverarbeitet.

Die IT-Infrastruktur besteht aus mehreren HP PA-RISC und Intel x86 Servern, auf denen die SAP R/3 Basiskomponenten und Applikationsserver ausgeführt werden. Neben den Standardkomponenten besteht P@RIS auch aus mehreren eigenentwickelten SAP R/3 ABAP Applikationen der SBS.

Die Erstellung ging von der bereits existierenden Infrastruktur von SBS aus und erfolgte auf Basis der existierenden Dokumentation von P@RIS. Bei der Erstellung des Modells wurden einige der bereits aufgeführten Probleme deutlich. SBS verfügt über eine sehr umfassende Modellierung der Geschäftsprozesse in ARIS. Die entsprechenden EPKs konnten direkt in das

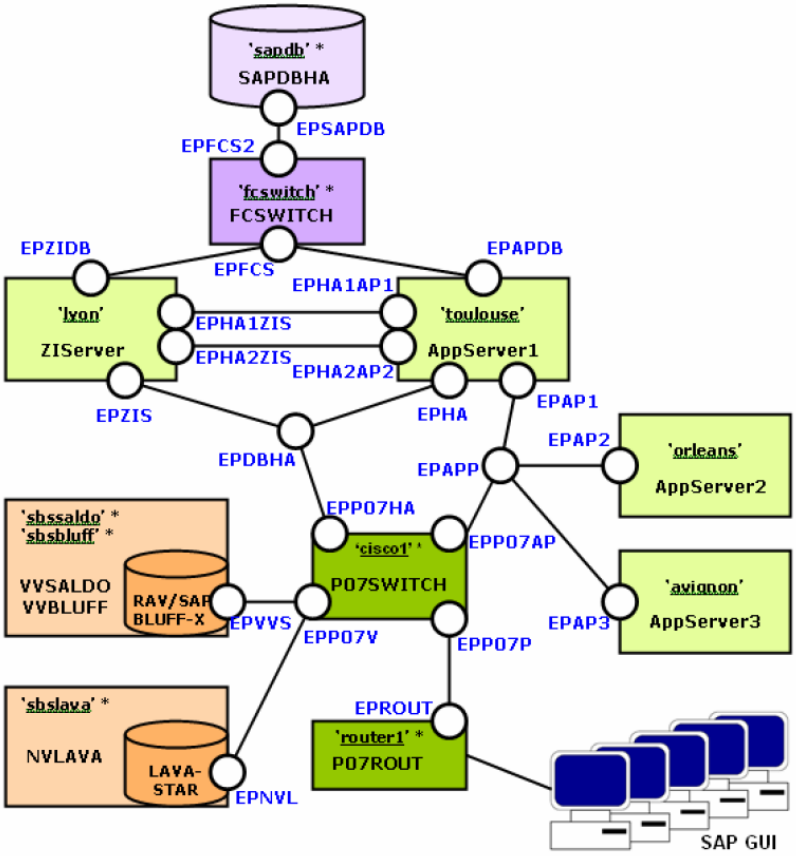

Abbildung 79 – Infrastruktur von P@RIS [Kunze06]

Gesamtmodell integriert werden. Die Erstellung eines eigenen Geschäftsprozessmodells konnte entfallen. Es wurden direkt die in den EPKs modellierten Funktionen als Basis für die Abbildung zwischen den Geschäftsprozessen und den Softwarekomponenten verwendet.

Das Softwaremodell war deutlich aufwendiger zu erstellen. Dies wurde durch die Struktur von SAP R/3 noch deutlich erschwert, das eher eine monolithische Anwendung als ein verteiltes System darstellt. Deswegen ist das Softwaremodell sehr kompakt.

Die Abbildung der Geschäftsprozesse auf die IT-Infrastruktur war bei SBS nicht vorhanden und war auch den Prozessverantwortlichen bei SBS nicht klar. Verantwortlich dafür ist vor allem die Aufgliederung der Verantwortlichkeiten bei SBS. Die Applikationsverantwortlichen sind dabei im Sinne des Applikationsstacks erst für die Anwendungsebene verantwortlich. Die Bereitstellung der IT-Ressourcen wie Serverinfrastruktur und Netzwerk erfolgt durch den IT-Betrieb im Re-

chenzentrum. Dazu zählt auch die Bereitstellung des Betriebsystems und von Basisdiensten für das Management der Server, wie die Bereitstellung der Hochverfügbarkeitslösung und das Backup des Systems.

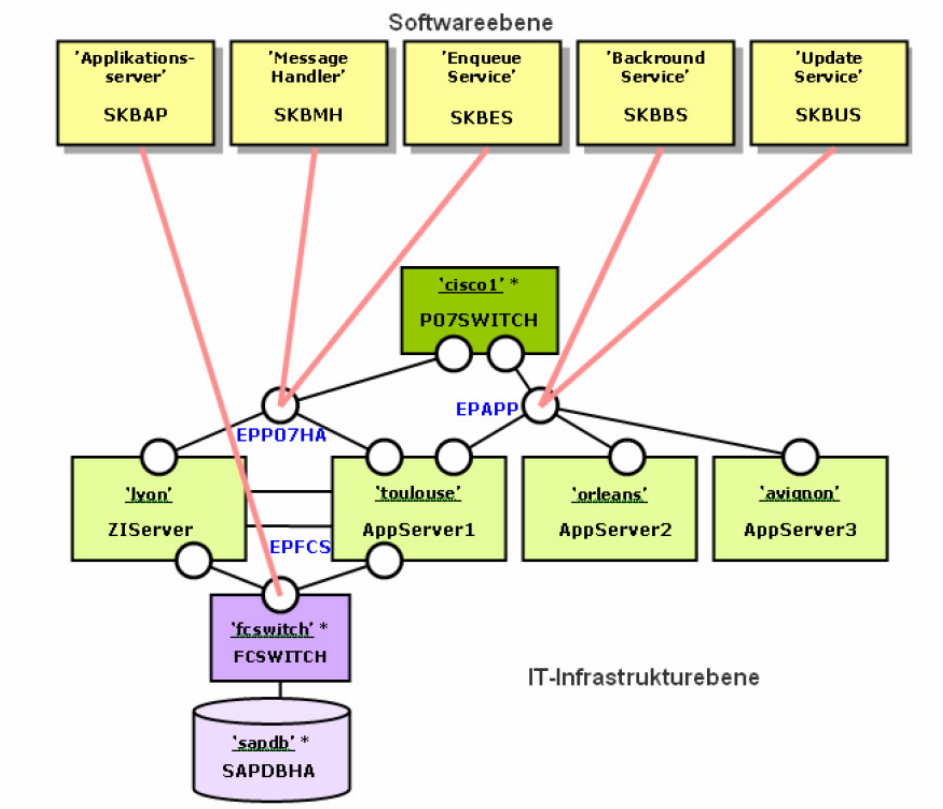

Abbildung 80 – P@RIS - Abbildung Softwarekomponenten auf IT-Infrastruktur [Kunze06]

Diese Informationen mussten in der Diskussion mit den verschiedenen Verantwortlichen bei SBS erarbeitet werden. Eine Spezialität der Infrastruktur von P@RIS ist der Einsatz einen prorietären Hochverfügbarkeitslösung für die beiden großen Serversysteme. Auch diese konnte mit dem Modell umfassend modelliert werden. Die Abbildung 79 verdeutlicht die Infrastruktur mit den modellierten Endpunkten.

Das IT-Infrastrukturmodell von P@RIS musste manuell erstellt werden, da die Einführung der CMDB bei SBS noch nicht in allen Rechenzentren abgeschlossen ist. Entgegen kam der Erstellung des Infrastrukturmodells hier die Tatsache, dass es sich bei P@RIS um eine Silo Installation handelt, bei der auf den vier betroffenen Servern nur diese Software ausgeführt wird. Darüber hinaus ist P@RIS wie viele andere Silosysteme voneinander durch einen Router abgeschottet. Dies führt dazu, dass die Grenzen der Infrastruktur sehr einfach modelliert werden konnten. Die Abbildung 80 verdeutlicht die Verteilung der Software auf den Serversystemen. Auch hier ist die Tatsache, dass es sich bei SAP R/3 eher um eine monolithische Applikation handelt, als Vorteil zu betrachten, der die Modellierung vereinfacht. Eine hochgradig verteilte Applikation mit vielen verschiedenen, voneinander unabhängig und in der Verteilung dynamischen Komponenten hätte die Modellerstellung deutlich erschwert.

Gerade das Interesse der verschiedenen Beteiligten an dieser Diplomarbeit und das Feedback bei der Präsentation bei SBS zeigen, dass dieser Bereich für den praktischen Einsatz eine große Relevanz hat. Die Verbesserung der Modelle der IT-Dienste trägt unter anderem zu einem besseren Verständnis, zum einfacheren Management von Veränderung und letztlich zu einer Optimierung der Kosten bei. Aus der Diplomarbeit haben sich Diskussionen mit SBS ergeben, auf die wir im Ausblick noch eingehen.

9.4 SBS Security Entscheidungstool

Im Rahmen der CKI Initiative wurde am Lehrstuhl von Prof. Brügge das Decision Support Tool entwickelt, mit dem die Bewertung von Sicherheitsvorfällen möglich wird [AJ06]. Verschiedene Arten von Angriffen werden modelliert und ihre Auswirkungen auf einen IT-Dienst analysiert.

Das Ziel des Projektes ist ein sogenanntes Cybernetic Defence System. Das System soll dabei Angriffe auf die Infrastruktur analysieren und bewerten, ob der finanzielle Schaden durch den Angriff größer ist, als das Abschalten des betroffenen Dienstes und diese Informationen dem Administrator zur Verfügung stellen.

In diesem Prototyp wurde unser IT-Dienstmodell als Basis für die IT-Dienste implementiert und die Modellierung von P@RIS verwendet. Bereits bei der Erstellung der DA hat es sich als schwierig erwiesen, einen Ausfall von P@RIS monetär zu bewerten. Dies erscheint in diesem Zusammenhang die schwierigste Aufgabe zu sein. Das Modell konnte in das Tool problemlos integriert werden, der Prototyp steht für Evaluierungen zur Verfügung.

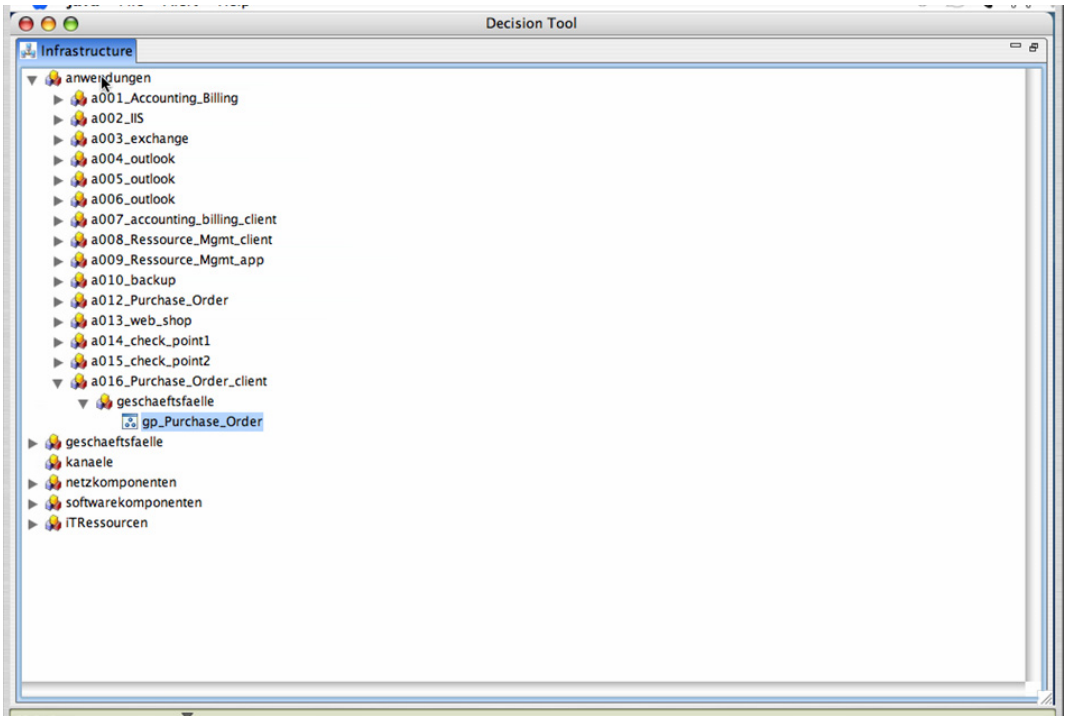

Abbildung 81 – Decision Support Tool [AJ06]

9.5 Fazit der Evaluierung

Das Modell und die Konzepte sind in dieser Form umsetzbar. Die Prototypen belegen die einzelnen Ideen, sie ermöglichen die Übernahme von Wissen aus der Softwareentwicklung und unterstützten den Betrieb der IT-Dienste.

Der IT-Modeller zeigt, dass eine Umsetzung in intuitiv bedienbares Modellierungswerkzeug möglich ist. Der Einsatz des Tools im Decision Support Tool ist ein weiteres Beispiel für den Einsatz und die Erweiterbarkeit des Modells. Das Tool belegt, dass das Modell auch zur Bewertung von IT-Diensten und der verwendeten Infrastruktur eingesetzt werden kann.

Die Evaluierung mit SBS belegt auch die praktische Relevanz des Modells und das fachlich Verantwortliche auf den jeweiligen Ebenen das Modell als geeignet erachten.

Wir werden im letzten Kapitel einen Ausblick geben, welche weiteren Forschungsthemen sich aus dem Modell ergeben.

10 Fazit und Ausblick

Unser vorgestelltes Modell beschreibt IT-Dienste mit all ihren verschiedenen Ebenen. Die Modellierung schlägt die Brücke zwischen der technischen und der geschäftlichen Sicht auf den IT-Dienst. Die Diplomarbeit [Kunze06] hat mit der Evaluierung am Beispiel von SBS P@RIS gezeigt, dass das Modell einsetzbar ist und reale Geschäftsprozesse abbilden kann. Darüber hinaus wurde die Integration existierender Modelle wie ARIS EPKs illustriert und die Möglichkeit der Integration so bewiesen.

Die Verbindung zwischen Entwicklung und Betrieb wurde durch die Prototypen bereits exemplarisch umgesetzt. Aktuelle Entwicklungen unter anderem im Umfeld von Microsoft Visual Studio 2005 lassen erahnen, dass dieser Punkt auch von den Herstellern von Entwicklungswerkzeugen aufgegriffen und in Managementwerkzeuge wie Microsoft SMS v4 integriert wird.

10.1 Aktueller Stand

Im Umfeld des Managements von IT-Diensten gibt es momentan verschiedene Ansätze, gerade Application Scanner sind im praktischen Einsatz ein bevorzugtes Mittel, da sie die Erstellung von Modellen vereinfachen. [NDVG06] verdeutlicht hier noch genauer die Unterschiede und die Einordnung der unterschiedlichen Konzepte. Unser Modell kann auch in diesem Umfeld seinen Beitrag leisten und die Schwächen, die Application Scanner haben, zumindest teilweise aufheben, indem es die Modellierung von nicht erkennbaren Komponenten und Systembestandteilen erlaubt. Darüber hinaus erscheint uns die Kopplung mit einem Incident Management Werkzeug als sehr interessant.

Dieses Modell wurde im Rahmen der Siemens SBS TUM Kooperation CKI – Dynamic Value Webs for IT-Services entwickelt. Präsentationen und Diskussionen mit verschiedenen Abteilungen bei Siemens haben gezeigt, dass ein entsprechendes Modell den täglichen Arbeitsablauf deutlich unterstützen würde. Das Modell wurde durchwegs positiv aufgenommen und die Entwicklung und der Einsatz eines entsprechenden Tools begrüßt.

Im Umfeld der Enterprise Architecture, sowie der Softwarekartographie mit ihren IT-Karten und Bebauungsplänen gibt es momentan mehrere kommerzielle und wissenschaftliche Ansätze, die ähnliche Zielsetzungen wie diese Arbeit verfolgen.

Die wichtigen Unterschiede sind dabei die gewünschten Ergebnisse. Während die von Garnter untersuchten Werkzeuge zur Modellierung von Enterprise Architecture Umgebungen stark auf die Beschreibung von Landschaften abzielen, hat die Softwarekartographie den weiteren Fokus in Richtung Bebauungsplanmanagement und Entwicklung von IT-Landschaften. Diese Zielrichtung fokusiert dabei mehr die strategische Planung von Infrastrukturen und weniger die konkreten Fragestellungen im Management und Aufbau eines Datacenters.

Es zeigt sich ein Trend, dass die verschiedenen Ebenen aufeinander zugehen. Die ITIL-basierten Managementwerkzeuge integrieren immer mehr auch eine Businesssicht auf die Geschäftsprozesse, die Tools zur Geschäftsprozessmodellierung wie das ARIS-Toolset werden mit Managementsystemen kombiniert. Ein Beispiel für diesen Trend ist ARIS for SAP Netweaver. Mit dieser Integration werden die im ARIS Toolset modellierten Geschäftsprozesse auf die Softwarekomponenten von Netweaver abgebildet. Für Standardprozesse existiert beim Kunden so bereits die Modellierung zwischen den beiden Ebenen, das konkrete Deployment wird über die Managementwerkzeuge von Netweaver erkannt und integriert. Dies zeigt, dass unser integrativer Ansatz den richtigen Weg zeigt und von vielen beschritten wird. BMC hat eine Kooperation mit IDS Scheer bekannt gegeben [IDSBMC06], mit dem Ziel die Atrium CMDB mit dem ARIS Toolset zu integrieren. Auch hier wird derselbe Ansatz verfolgt, Teilmodelle zu integrieren. Das als herausragende Funktion in der Presseerklärung beschriebene Feature des Businessprozessmonitorings, das Ausfälle der IT-Infrastruktur auf Geschäftsprozesse abbildet, sollte allerdings kritisch betrachtet werden. Eine Evaluierung verschiedener CMDBs bei SBS hat einen deutlichen Unterschied zwischen den beworbenen Funktionen und dem tatsächlichen Produkt im täglichen Einsatz gezeigt, bei dem selbst die Integration mehrer Tools eines Herstellers gescheitert ist. Ähnliches gilt für Aussagen von Microsoft, die von einer 70% Automatisie-

rung der Umsetzung von ARIS Prozessen auf Biztalk 2006 Objekte sprechen [MSBT06]. Es zeigt aber, dass dieser Markt sehr stark in Bewegung ist.

10.2 Weitere Forschungsbereiche

Wir haben bereits bei den einzelnen Kapiteln Aspekte beschrieben, die die Arbeit nicht behandelt, die aber weiterer Forschung bedürfen. Wir wollen die Bereiche, die sich für weitere Forschungsarbeiten anbieten, kurz vorstellen:

- Entwicklung eines zeitlichen Referenzmodellsystem zur Deployment-Planung

  Wir haben beschrieben, dass das Modell zeitliche Einschränkungen und maximale Laufzeiten beim Deployment beachten kann, wenn es die zeitlichen Abläufe der einzelnen Komponenten kennt und diese von einem Referenzsystem auf die tatsächliche Infrastruktur übertragen kann. Dazu muss die Ausführungszeit auf dem Referenzsysstem bekannt sein und die Übertragung der Daten auf die produktive Umgebung möglich. Dabei muss sichergestellt sein, dass die Hardwarearchitekturen vergleichbar sind, oder diese Einflüsse müssen in die Berechnung mit einfließen.

  Hierzu ist die Entwicklung eines Referenzmodells und vor allem die Übertragbarkeit der Messergebnisse sicherzustellen. Beides könnte im Rahmen von weiteren Arbeiten entwickelt werden.

- Entwicklung eines Tools zum fachlichen Managements von IT-Diensten

  Wir haben bereits in der Einleitung unsere Vision zum Policy-based Management von IT-Diensten beschrieben. Wünschenswert wäre die Entwicklung eines Tools, das aus fachlichen SLA's die IT-Infrastruktur konfiguriert und deren Gesamtauslastung optimiert. Dieses Tools kann nur durch das Zusammenspiel vieler verschiedener Tools erreicht werden, wir haben im Abschnitt über ITIL Managementtools dazu bereits einige vorgestellt. Diese könnten eine Basis für eine prototypische Implementierung für ein Service Dashboard sein.

  Dazu ist vor allem die Komponente zur Optimierung wichtig, die wieder Anknüpfungspunkte zum vorigen Forschungsthema hat, auch hier ist die Vergleichbarkeit von IT-Architekturen ein wichtiger Aspekt.

- Integration der Konzepte der Softwarekartographie mit dem Modell

  Der Schwerpunkt der Arbeiten in den Projekten der Softwarekartographie am Lehrstuhl von Prof. Matthes liegt im Gegensatz zu dieser Arbeit auf einer höheren, strategischen Ebene. Während diese Arbeit ein Modell beschreibt, dessen Schwerpunkt auf der feingranularen Beschreibung von Abhängigkeiten liegt und das damit als Grundlage für ein umfassendes Managementwerkzeug dienen kann, liegt der Schwerpunkt bei der Softwarekartographie auf der Planung und der Weiterentwicklung von Bebauungsplänen. Der Fokus liegt hier wie schon beschrieben deutlich stärker auf der Architektur dieser Infrastrukturen.

  Gerade im Hinblick auf Service-orientierte Architekturen und deren Implementierungen wie Webservices erscheint es jedoch wichtig, auch die technische Realisierung nicht aus den Augen zu verlieren, da nicht alle Aspekte der IT-Infrastruktur zentral geplant werden und häufig Applikationen an der zentralen Steuerung vorbei „entstehen". Durch die Integration von entsprechenden Integrationsmöglichkeiten in Produkte wie Microsoft Office werden dabei neuen Benutzergruppen Möglichkeiten in die Hand gegeben, Softwaresysteme aufzubauen und dieser Aspekt eher verstärkt. Dieser Übergang hin zu einem Information Worker, der aus den entsprechenden Webservices Informationen bezieht, entzieht sich dem Ansatz einer zentralistischen Planung.

  Die Integration der Modelle der Softwarekartographie mit dieser Arbeit scheint ein interessanter Weg zu sein. Erste Planungen auf der Basis des IT-Modelers existieren hierzu bereits. Durch die grafischen Darstellungen und Bebauungspläne, die die Software-

kartographie definiert, entstehen hier interessante Erweiterungen für unser Modell und es entsteht eine noch detaillierteres Bild eines IT-Dienstes.

- Die monetäre Bewertung von IT-Diensten

  In einzelnen Brachen existieren bereits Teilmodelle zur monetären Bewertung von IT-Diensten, vor allem im Bereich der Banken und Versicherungen, die zu einem umfassenden Risikomanagement gezwungen sind. Die Verallgemeinerung dieser Modelle scheint viele neue Möglichkeiten zu bieten. Die Arbeit hat aber gezeigt, dass dies einen sehr umfassenden Ansatz benötigt und nicht trivial ist, da viel Experten- und Detailwissen notwendig ist.

- Security Modelling / Security Evaluation

  Unser Modell beinhaltet umfangreiche Informationen über die Dienste einer IT-Infrastruktur. Konzepte wie die Attack Graphs [JW03] erlauben eine Beschreibung von Angriffen auf IT-Systeme. Eine Kombination dieser beiden Techniken könnte zu einem Werkzeug führen, dass Modelle von IT-Infrastrukturen auf bekannte Sicherheitsprobleme testet.

  Aufbauend darauf könnte das Tool dazu verwendet werden, die Sicherheit in diesen Infrastrukturen zu verbessern, indem unbenutzte Dienste deaktiviert werden. Darüber hinaus könnten diese Informationen die Arbeitsweise von Intruder Detection Systemen verbessern, die heute vor allem das Problem haben, erlaubte von unerlaubten Zugriffen zu unterscheiden. Sind alle erlaubten Zugriffe durch ein IT-Dienstmodell beschrieben, entfällt hier die Lernphase für entsprechende Tools.

Es ergeben sich viele interessante Richtungen, die Erweiterungen darstellen oder unser Modell als Basis für neue Lösungen verwenden. Einige davon werden eventuell im Rahmen des CKI Projekts weiter verfolgt.

10.3 Fazit und Ausblick

Die Erstellung und die Arbeit an unserem IT-Dienstmodell zeigt, dass IT-Dienste immer komplexer werden. Es gibt immer mehr Einflüsse, die beachtet werden müssen, momentan vor allem technische Neuerungen und gesetzliche Vorgaben. Die verschiedenen Perspektiven und Menschen müssen hierbei ein gemeinsames Verständnis für den IT-Dienst entwickeln. Dazu leistet das Modell einen Beitrag, indem es diese Sichten integriert.

Im Rahmen des CKI Projektes wird in nächster Zeit der Einsatz des Modells im Umfeld der ITIL Tools CMDB und Application Scanner erfolgen, um verschiedene Sichten auf einen IT-Dienst zu integrieren. Getrieben wird diese Integration nicht nur alleine um die Prozessdokumentation zu verbessern, sondern unter anderem auch, um Kosten bei der Erbringung von Diensten zu sparen und die Erbringung der IT-Dienste zu verbessern.

Literaturverzeichnis (1)

[GU98] Die grenzenlose Unternehmung,
Arnold Picot, Ralf Reichwald, Rolf T. Wigand, Gabler, 1998

[IBM06] IBM Global CEO Study 2006, IBM, 2006

[Cro05] Deutscher Corporate Governance Kodex,
Regierungskommission Deutscher Corporate Governance Kodex, 2005

[BMW03] Warum BMW auf Storage on Demand abfährt,
Netzguide Outsourcing/Managed Services, Netzmedien AG, 2003

[Weill02] Don't just Lead, Govern: Implementing Effective Governance,
Weill/ Woodham, Cambridge: Sloan School of Management, 2002

[BITKOM05] Leitfaden E-MAIL-Archivierung,
Ralph Hintemann, BITKOM e.V., 2005

[AR01] ARIS, Modellierungsmethoden, Metamodelle, Anwendung,
August-Willhem Scheer, Springer, 2001

[ITIL] ITIL Website, www.itil.org

[WP-ITIL] ITIL, Wikipedia Website, http://de.wikipedia.org/wiki/IT_Infrastructure_Library

[Krcmar] Informationsmanagement,
Helmut Krcmar, Springer, 2004

[ISO2k] ISO 20000, Wikipedia Website, http://de.wikipedia.org/wiki/ISO_20000

[Broy, Stølen 01] Specification and Development of Interactie Systems: FOCUS on Streams,
Interfaces, and Rifnement, M. Broy, K. Stølen, Springer, München 2001

[ViSEK] Virtuelles Software Engineering Kompetenzzentrum Website,
www.software-kompetenz.de

[OMG] Object Management Group Website, www.omg.org

Literaturverzeichnis (2)

[WP-UML] UML, Wikipedia Website, http://de.wikipedia.org/wiki/Unified_Modeling_Language

[GI-PN] Petrinetze, GI Fachgrupe Petrinetze Website,
http://www.informatik.uni-hamburg.de/TGI/GI-Fachgruppe0.0.1/PN.html

[Pag06] ASP.net - ISA 2004/2006 Integration,
Dennis Pagano, TU München, München 2006

[Thu04] Formal fundierte Modellierung von Geschäftsprozessen,
Veronika Thurner, TU München, München 2004

[Tan03] Computernetzwerke
Andrew S. Tannenbaum, Prentice Hall, München 2003

[Daw05] Real-Time Enterprise Demand Real-Time Infrastructures,
Philip Dawson, Gartner Sympsoium ITXPO, Barcelona 2005

[Heg99] Integriertes Management vernetzter Systeme,
Heinz-Gerd Hegering, Sebastian Abeck, Bernhard Neumair, dpunkt Verlag, München 1999

[Broy98] Informatik, eine grundlegende Einführung,
Manfred Broy, Springer Verlag, Berlin 1998

[Thiele05] IT-Bebauungsmanagement, Prozess- und Datenmodell,
Anja Thiele, TU München, 2005

[Zarn05] Die Sorgfaltspflicht der IT – IT-Service-Management und IT-Governance,
Dr. Rüdiger Zarnekow, Vortrag Universität St. Gallen, 17.Mai 2005

[Sym05] IT Governance Framework – Structures, Processes and Communication,
Craig Symons, Forrester Research, 2005

[Kunze06] Analyse und Modellierung von Geschäftsprozessen im Rahmen des Projektes SBS SAP Konsolidierung, Katja Kunze, TU München, 2006

[Mercury06-1] Mercury Application Mapping Produktdokumentation,
Mercury Interactive Cooperation

Literaturverzeichnis (3)

[Mercury06-2] Mercury CMDB / MAM, Generische Betrachtung zu Wertschöpfungspotentialen, Jürgen Basten, Mercury (Interne Präsentation)

[NDVG06] Vergleich zwischen einem Application Scanner und dem Modell für verwaltbare IT-Dienste, Norbert Diernhofer, TU München, München 2006

[MOF] Microsoft Operations Framework Website, www.microsoft.com/mof

[MSEX04] Microsoft TechNet Roadshow 2004,
Björn Schneider, Microsoft Deutschland GmbH, München 2004

[Basel2] Basel II, Website der Bundesbank,
http://www.bundesbank.de/bankenaufsicht/bankenaufsicht_basel.php

[HBR04] IT Doesn't Matter,
Nicolas Carr, Harvard Business Review, 2004

[CKI] CKI Website TUM, http://portal.mytum.de/cki/index_html

[OABPEL] OASIS WSBPEL Website,
http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=wsbpel

[DMTF] CIM Website DMTF, http://www.dmtf.org/standards/cim/

[WP-BPEL] Wikipedia BPEL, http://de.wikipedia.org/wiki/BPEL

[WP-ITCont] Wikipedia IT Controls, http://en.wikipedia.org/wiki/Information_technology_controls

[AUDIRIM] Wirft Audi seine Blackberrys raus,
Artikel Computerwoche online, http://www.computerwoche.de/index.cfm?pid=380&pk=557254

[XEN03] Xen and the Art of Virtualization,
Paul Barham et. Al., University of Cambridge, 2003

[GOF95] Design Patterns. Elements of Reusable Object-Oriented Software,
Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides, Addison Wesley, 1995

Literaturverzeichnis (4)

[AJ06] Decision Support Tool,
Lilith Al-Jadiri, Präsentation SBS, 2006

[IDSBMC06] BMC Software und IDS Scheer stellen gemeinsame BSM-Lösung, http://www.ids-scheer.com/germany/consulting/87669 Presseerklärung IDS Scheer, 2006

[CobiT4] CobiT Framework, IT Governance Institute, www.itgi.org

[ITIL-Web] Website ITIL, www.itil.org

[MSBT06] Microsoft Biztalk 2006,
http://www.microsoft.com/germany/biztalk/interop/prozessmodellierung.mspx

[JW03] Vulnerability Analysis Using Attack Graphs,
Jeannette Wing, MSR Academic Conference Germany 2003, Dresden

Index (1)

Index (2)