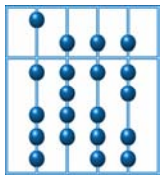
Technische Universität München
Fakultät für Informatik



Diplomarbeit

Integrierte Modelle und Sichten für das IT-Management

Analyse und Entwicklung in Zusammenarbeit mit der HVB Systems

Katharina Brendebach

Aufgabensteller:	Prof. Dr. Florian Matthes
Betreuer:	Peter Ehinger, HVB Systems
	André Wittenburg, TU München
Abgabedatum:	15. Juli 2005

Ich versichere, dass ich diese Diplomarbeit selbständig verfasst und nur die angegebenen Quellen und Hilfsmittel verwendet habe.

Markt Indersdorf, 04.07.2005

Katharina Brendebach

Danksagung

Ich möchte mich bei Prof. Dr. Florian Matthes bedanken, der mir die Gelegenheit gab mich intensiv mit dem Thema IT-Management auseinander zusetzten. Ich empfand die mir gestellte Aufgabe als eine besondere Herausforderung, die mir den Einblick in ein überaus interessantes Themengebiet ermöglichte.

Ich bedanke mich bei Peter Ehinger, der mich hervorragend in das Team und das Unternehmen integrierte. Seine wertvollen Anregungen und seine investierte Zeit, sowie die Kontakte zu zahlreichen Mitarbeitern der HVB Systems und HVB Info, trugen zum Gelingen dieser Arbeit bei.

André Wittenburg danke ich für eine außerordentliche, stets präsente Betreuung der ganzen Arbeit, insbesondere bei der Anfertigung der Ausarbeitung. Der Gedankenaustausch mit ihm eröffnete mir immer wieder neue Sichten auf das Thema.

Mein besonderer Dank gilt meinen Eltern und meinem Partner, die mich stets in meinem Studium unterstützt haben.

Markt Indersdorf, 15.07.2005

Katharina Brendebach

Zusammenfassung

Die Informationstechnologie stellt für heutige Unternehmen einen wichtigen Produktions- und Wettbewerbsfaktor dar. Das IT-Management verlangt aus diesem Grund eine integrative Sicht auf die Unternehmensarchitektur, um die Informationstechnologie an den Bedürfnissen des Geschäfts auszurichten und den Gestaltungsprozess der IT-Landschaft geeignet steuern zu können.

Die Arbeit stellt ein integriertes Informationsmodell der Unternehmensarchitektur vor, das dazu dient, relevante Objekte, deren Attribute sowie Beziehungen zu identifizieren und abzubilden. Die Analyse vorhandener Modelle und Karten der HVB Systems lieferte dabei erste Hinweise für Bestandteile des Modells.

Unter zu Hilfenahme des Informationsmodells sowie den Modelle und Methoden des Forschungsprojektes Softwarekartographie, wurden neue Sichten für das IT-Management entwickelt, um die Architektur wahrnehmbar und erfahrbar zu machen. Die Visualisierung der im Modell strukturierten Informationen mittels geeigneter graphischer Notationen ermöglicht auf diese Weise einen verbesserten Informationstransfer zu den Stakeholdern, indem ihre Interessen und Fragestellungen geeignet adressiert und beantwortet werden. Eine exemplarische Umsetzung im konkreten Einsatzszenario der HVB Systems untersucht die entwickelten Darstellungen auf ihre Eignung.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Kapitel 1
Einleitung

Gegenwärtig findet ein Wandel von einer Dienstleistungsgesellschaft in eine Informationsgesellschaft statt, d.h. der Grad der Informiertheit eines Unternehmens bestimmt seine Stellung auf dem Markt [GA02]. Da Informationen zur unternehmerischen Ressource schlechthin zählen, avanciert die IT laut Gernert et al. [GA02] zu einem der wichtigsten Produktions- und Wettbewerbsfaktoren in modernen Unternehmen. In einigen Branchen, wie beispielsweise dem Finanzdienstleistungssektor, kann sie sogar zum kritischen Erfolgsfaktor zählen [GA02]. Zusätzlich werden laut Esser [Es02] „(…) die Innovationszyklen von Produkten und Dienstleistungen immer kürzer, so dass die informationelle (…) Anpassungs- und Reaktionsfähigkeit für Unternehmen zunehmend zu Existenzsichernden Eigenschaften werden."

Da sich die Informationstechnologie in den letzten Jahren grundlegenden Veränderungen unterzog, führt die historisch gewachsene, sehr heterogene Prozess- und IT-Landschaft in vielen Unternehmen durch ihre enorme Komplexität zu einer hohen Kostenbelastung und einer geringen Flexibilität[1]. Fehlende Standardisierung, wenig Integrationstiefen und fehlende Transparenz fordern nach Instrumenten, die dieser oftmals als Wildwuchs bezeichneten Situation entgegen wirken. Kosten müssen gedrückt sowie neue fachliche Anforderungen schneller umgesetzt werden [BH04].

Das Konzept der Unternehmensarchitektur soll durch eine gesamthafte Betrachtung des Unternehmens und der IT, zur geeigneten Steuerung des Gestaltungsprozesses der IT-Landschaft beitragen. Soll die Architektur dabei wahrnehmbar und erfahrbar gemacht werden, so bedarf es Modellen und integrativen Sichten selbiger, die der Dokumentation dienen und die Kommunikation zwischen Menschen ermöglichen.

Die Arbeit entsteht im Rahmen des Forschungsprojektes Softwarekartographie des Lehrstuhls für Software Engineering betrieblicher Informationssysteme der TU München sowie in Zusammenarbeit mit dem Projektpartner HVB Systems, einer hundertprozentigen Tochter der Bayerischen Hypo- und Vereinsbank AG. Erkenntnisse und erarbeitete Methodiken aus dem Forschungsprojekt, dass sich mit der Entwicklung von Methoden und Modellen zur Beschreibung, Bewertung und Gestaltung von Anwendungslandschaften beschäftigt, sollen in die Wirtschaft übertragen und somit praxisnah erprobt werden.

1.1 Aufgabenstellung

Die Herausforderung der vorliegenden Arbeit liegt in dem Aufbau eines integrierten[2] Informationsmodells, das die Struktur für alle benötigten Daten der Unternehmensarchitektur vorgibt, die dann in unterschiedlichen erforderlichen Sichten für das IT-Management visualisiert werden. Die Visualisierung mittels geeigneter graphischer Notationen soll einen verbesserten Informationstransfer zu den Stakeholdern ermöglichen, indem Interessen und Fragestellungen der Stakeholder geeignet adressiert und beantwortet werden.

Das Informationsmodell soll aufbauend auf der existierenden IT-Architektur der HVB Systems, sowie den vorhandenen Modellen und Karten entwickelt werden. Weitere Anforderungen die bisher nicht abgebildet werden konnten, sollen das entwickelte Modell ergänzen und abrunden. Begleitend soll ein zu definierender Begriffsapparat ein einheitliches Verständnis im Umfeld der IT-Architektur fördern. Eine exemplarische Umsetzung im konkreten Einsatzszenario der HVB Systems soll die entwickelten Darstellungen auf ihre Eignung hin untersuchen.

---

[1] Vgl. System, Komplexität, Kompliziertheit und Flexibilität in [Es02].

[2] Das Informationsmodell soll eine einheitliche, umfassende Beschreibung der Architektur in einer Modellierungssprache bieten. Es soll nicht durch die Verknüpfung von einzelnen Modellen unterschiedlicher Methoden gebildet werden [vgl. Mü98].

1.2 Gliederung

Im zweiten Kapitel *Einbettung in das thematische Umfeld* erfolgt eine Einführung in den Kontext dieser Arbeit. Die Erläuterung der Begriffe IT-Management, Unternehmensarchitektur und Softwarekartographie sollen die Motivation und Aufgabenstellung der Arbeit festigen.

Eine Heranführung an die für das Verständnis dieser Arbeit benötigten Themengebiete, wird durch das dritte Kapitel der *theoretischen Grundlagen* ergänzt. Begriffe rund um das Thema Modell sowie für die Modellierung benötigte Konzepte werden vorgestellt.

Die *Analyse der bestehenden Diagramme und Karten* in Kapitel vier vertiefen die Konzepte und Problemstellungen des zweiten und dritten Kapitels und liefern erste Sichten für das IT-Management.

Die durch die Analyse eingeleitete Identifikation von Objekten, Attributen und Beziehungen wird im fünften Kapitel *Entwicklung eines integrierten Informationsmodells* fortgesetzt und stellt den zentralen Teil der Arbeit dar.

Eine exemplarische Umsetzung von Teilbereichen des Informationsmodells erfolgt innerhalb des sechsten Kapitels *Umsetzung von Teilaspekten des Informationsmodells.*

Abschließend werden im siebten Kapitel *Zusammenfassung und Ausblick* die Ergebnisse und Erfahrungen der Arbeit zusammengefasst, sowie Ausblicke auf weiterführende Themen aufgezeigt.

Werden die Kapitel nach den Meilensteinen eines Vorgehensmodell für das Software Engineering untersucht, so stellt die Analyse der bestehenden Diagramme und Karten die *Analyse*, die Entwicklung des Informationsmodells das *Design* und das Kapitel zur Umsetzung von Teilmodellen die *Implementierung* dar.

Kapitel 2

Einbettung in das thematische Umfeld

In diesem Kapitel soll in das thematische Umfeld der Arbeit durch die Vorstellung der Themengebiet IT-Management (Abschnitt 2.1), Unternehmensarchitektur (Abschnitt 2.2) und Softwarekartographie (Abschnitt 2.3) eingeführt werden.

2.1 IT-Management

Die Arbeit trägt den Titel *Modelle und Sichten für das IT-Management*. Es stellt sich somit zunächst die Frage, was unter dem Begriff IT-Management zu verstehen ist. Laut Bernhard et al. [BBB03] ist das IT-Management eine „(…) Managementaufgabe, die den Geschäftserfolg direkt beeinflusst, indem sie den Wertbeitrag der IT zum Unternehmenserfolg steigert und gleichzeitig die mit der IT verbundenen Risiken und Kosten minimiert".

Die Stellung des IT-Managements innerhalb des Unternehmens wächst gegenwärtig stark, da sich die Informationstechnologie (IT) in jüngerer Zeit von einer einfachen Ressource zu einem strategischen Erfolgsfaktor entwickelt hat [BBB03]. Laut Gernert et al. [GA02] ist sie sogar einer der wichtigsten Produktions- und Wettbewerbsfaktoren des Unternehmens. Nach Buchta et al. [BES04] kann die IT sogar als *Enabler*[3] eines Unternehmens bezeichnet werden, wenn das Geschäft ohne die IT nicht abgewickelt werden kann (z.B. Online banking).

An die informationstechnischen Mittel werden immer vielfältigere Anforderungen gestellt. Es ist zu beobachten, dass der entstehende Aufwand in der heutigen IT, gegenüber dem vor 10 Jahren, gestiegen ist. Ein leistungsfähiges und umfassendes IT-Management ist erforderlich, um unter Berücksichtigung der Ressourcen des Unternehmens und den vielfältigen Möglichkeiten der modernen Informationstechnologien, ein optimales *Alignment*[4] von Geschäft und IT zu erzielen [GA02]. Unter dem Begriff *IT-Alignment* lässt sich demnach die Ausrichtung der IT an der Unternehmensstrategie, den Unternehmensprozessen und der Unternehmenskultur verstehen [MS03].[5]

Wird das IT-Management anhand seiner zwei Bestandteile zerlegt und näher spezifiziert, so umfasst der Begriff *IT* alle technischen und organisatorischen Mittel zur Unterstützung der Beschaffung, Verarbeitung, Speicherung, Übertragung und Bereitstellung von Informationen [GA02]. Der Begriff *Management* beinhaltet die Festlegung der Ziele und Strategien für das Unternehmen, deren Operationalisierung und Überwachung, sowie die Entwicklung einer Unternehmensorganisation. Das Management muss zur Erreichung der Ziele Mitarbeiter führen und motivieren, den Unternehmensprozess steuern, sowie Entscheidungen treffen [GA02]. Demgemäß ist das IT-Management nach Gernert et al. [GA02] „(…) die Wahrnehmung der üblichen Managementaufgabe konkret bezogen auf den Gegenstand IT."

Das IT-Management steht sehr stark in Zusammenhang mit dem Informationsmanagement, da es eine Vielzahl von Informationen benötigt und selbst neue Informationen generiert. Es ist jedoch eindeutig zwischen diesen beiden Managementaufgaben zu unterscheiden, da das IT-Management nur die Teile des Informationsmanagement betrachtet, die elektronisch unterstützt werden [GA02]. Als Informationsmanagement wird jedoch das Management in einem Unternehmen in Bezug auf Information und Kommunikation bezeichnet, dass alle zugehörigen Führungsaufgaben umfasst [He02]. Das Informationsmanagement betrachtet Informationen im Unternehmen demzufolge zunächst völlig unabhängig davon, ob sie elektronisch erzeugt, verarbeitet oder verwaltet werden [Ga02].

Die Aufgabe des IT-Managements besteht in der Gewährleistung einer ganzheitlichen Sicht auf die IT im Unternehmen, die zum einen die IT-Ziele, Strategien und Maßnahmen aufeinander abstimmt und zum anderen ein einheitliches Vorgehen für die Zielfindung, Planung, Überwachung und Steuerung für

[3] Von *to enable*, deutsch befähigen.

[4] Deutsch Anpassung, Ausrichtung.

[5] Laut einer Studie der Society for Information Management (SIM) im Sommer 2003, stellt das „IT and business alignment" das wichtigste Managementinteresse der IT Führungskräfte dar [EL04].

alle IT-Prozesse in allen Bereichen des Unternehmens sicherstellt. Unternehmens- und IT-Management müssen miteinander verbunden werden, damit sie bei der Zielformulierung und Umsetzung eng zusammenarbeiten. Nur auf diese Weise kann das Unternehmen so organisiert werden, dass es dynamisch und stabil zugleich ist und wettbewerbsfähig am Markt bestehen kann [GA02].

Des Weiteren lässt sich das IT-Management in einen strategischen und einen operativen Aspekt teilen. Beim strategischen IT-Management liegt der Fokus im Gegensatz zum operativen IT-Management nicht auf der Implementierung, sondern es muss entscheiden, wie durch den Einsatz von Informationstechnologie Wert für das Unternehmen geschaffen werden kann [BES04]. Das strategische Management fragt nach den richtigen Dingen die gemacht werden sollen, das operative Management beschäftigt sich mit der Frage, ob die Dinge richtig ausgeführt werden [GA02].

In der vorliegenden Arbeit sollen Konzepte und Werkzeuge erarbeitet werden, die überwiegend das strategische IT-Management in seinen drei Imperativen unterstützt: Erstens müssen Angriffspunkte gefunden werden, durch welche die IT die Unternehmensstrategie unterstützt bzw. mitgestaltet und somit den Wert des Unternehmens steigert. Zweitens muss die Leistung der IT geeignet gesteuert werden können und drittens sollen Bereiche identifiziert werden, in denen IT-Optimierungen vorgenommen werden und somit Kosten gesenkt werden können [BES04].

Aufgaben und Tätigkeiten des IT-Managements werden häufig auf konkrete Teilbereiche der IT bezogen, deren Begrifflichkeiten an dieser Stelle angeführt werden sollen (siehe Abbildung 1). Ist die Sprache von der so genannten *IT-Landschaft* des Unternehmens, so versteht man darunter die Gesamtheit aller IT-Systeme. Letzteres stellt dabei die Gesamtheit aller informationstechnischen Mittel und der dafür benötigten Organisationsumgebungen dar, die zur Unterstützung der Geschäftsprozesse bereitgestellt werden [GA02]. Da sich ein IT-System aus mehreren IT-Komponenten[6] zusammensetzt [GA02], lässt sich die IT-Landschaft in die *Software*- und Hardwarelandschaft unterteilen. Erste umfasst alle Softwaresysteme des Unternehmens, bei denen es sich um betriebliche Informationssysteme oder auch Applikationsserver bzw. Dateiserver etc. handeln kann. Die *Hardwarelandschaft* beinhaltet dahingegen alle physischen Komponenten zur Unterstützung der Informationssysteme [MW04a].

Die Anwendungssysteme bzw. betrieblichen Informationssysteme des Unternehmens stehen häufig im Mittelpunkt des IT-Managements, so dass ihre Gesamtheit unter dem Begriff der Anwendungslandschaft zusammengefasst wird [MW04a].

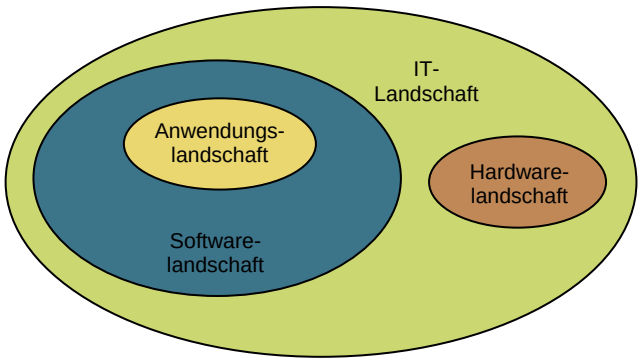


Abbildung 1: Abgrenzung relevanter Begriffe nach [Se05]

Soll das IT-Management durch den Einsatz von Informationstechnologie Wert für das Unternehmen schaffen [BES04], so bedarf es geeigneter Instrumente, um vor allem die Effektivität zukünftiger IT-Vorhaben beurteilen zu können [BBB03]. Eine Strukturierung und Identifizierung der beteiligten Komponenten und Objekte der IT-Landschaft, sowie die Integration der betrachten Sachverhalte ist dabei unabdingbar, um das Nutzenpotential der IT im Sinne des Unternehmens vollständig auszuschöpfen [BES04]. Im folgenden Abschnitt soll das Konzept der Unternehmensarchitektur eingeführt werden, die diesen Anforderungen gerecht wird und mehr und mehr an Bedeutung gewinnt [La05b].

[6] Hardware-, Software- oder Organisationskomponenten.

2.2 Unternehmensarchitektur

Ein wesentlicher Bereich des IT-Managements liegt in der Gestaltung und Bereitstellung einer optimalen Unternehmensarchitektur. Um diesen Begriff zu erläutern, bedarf es zunächst der Hinterleuchtung des Begriffes Architektur.

Nach der *Encyclopaedia Britannica* wird Architektur verstanden als: “Art and technique of designing and building, as distinguished from the skills associated with construction.“ [Enc04]. Auch in der Definition des IEEE Standards 1471-2000 (kurz IEEE 1471) ist die Sprache von dem Design. Eine Architektur ist hier: „The fundamental organization of a system embodied in its components, their relationships to each other, and to the environment, and the principles guiding its design and evolution.“ [IE00]

Letztere Definition findet in der Open Group Architekturdefinition Verwendung, die jedoch um zwei weitere, die kontextabhängige Verwendung betreffende Bedeutungen ergänzt wird [TOG05]. Demgemäß kann die Architektur einerseits als eine formale Beschreibung bzw. einem detaillierten Plan eines Systems auf Komponentenebene gesehen werden, um dessen Implementierung zu lenken. Andererseits veranschaulicht eine Architektur die Struktur der Komponenten, dessen Zusammenhänge und Prinzipien bzw. Richtlinien um dessen Design und Evolution über die Zeit zu steuern [TOG05].

Das gemeinsame Verständnis der genannten Definitionen liegt darin, dass eine Architektur ein System und dessen Bestandteile beschreibt. Wird der Begriff der Architektur nun auf ein Unternehmen bezogen, so wird das Unternehmen selbst als ein System[7] aufgefasst. Es interessiert eine ganzheitliche Sicht auf das Unternehmen, die sich durch vielfache Teilsysteme und mehrere funktionelle Gruppen des Unternehmens zieht [TOG05]. Laut A. Lapkin berührt sie sogar jeden Bestandteil der Organisation [La05b].

Ross [Ro03] definiert den Begriff *IT-Architecture* wie folgt: „(…) At the enterprise level, an IT architecture is the organizing logic for applications, data, and infrastructure technologies, as captured in a set of policies and technical choices, intended to enable the firm’s business strategy.“ Die Herausforderung eines Unternehmens liegt demnach in dem Erreichen eines ausgeglichenen Zusammenspiels von Geschäft und IT, indem sich beide Parteien gegenseitig unterstützen (siehe Abbildung 2). Der Begriff der *IT-Architecture* nach Ross [Ro03] wird in dieser Arbeit dementsprechend um geschäftliche Komponenten hin zur Unternehmensarchitektur erweitert. Voraussetzend müssen dabei Geschäftstrategie und -ziele sowie Geschäftsprinzipien von der IT verstanden werden, wobei zusätzlich eine zyklische Synchronisation von Geschäfts- und IT-Strategie unabdingbar ist [Ro03].

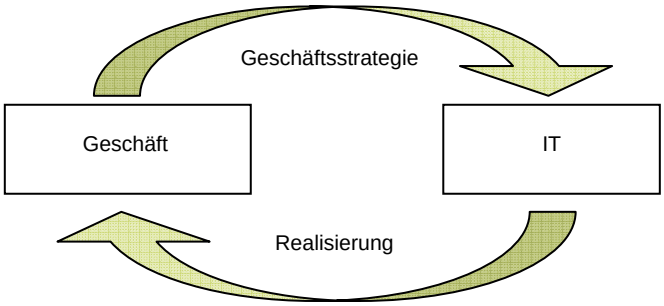


Abbildung 2: Zusammenspiel von Geschäft und IT angelehnt an [HVB04b]

[7] Ein System ist eine Menge von Komponenten, die miteinander in Beziehung stehen, um eine bestimmte Zielsetzung zu erfüllen, und die sich klar von ihrem Umfeld abgrenzen lassen. Ein System wird definiert über seine Zielsetzung, Grenzen, Komponenten, Struktur und Verhalten [GA02]. Der IEEE Standard 1471 definiert das System als: „A collection of components organized to accomplish a specific function or set of functions“ [IE00]. In der Systemtheorie wird beispielsweise das Beziehungsgefüge, d.h. die Verbindung der Systemelemente hervorgehoben, wobei die Komplexität und die Beherrschung von Komplexität zum zentralen Gegenstand wird [Es02].

Die Balance zwischen der IT-Wirksamkeit und des Geschäfts wird laut der Open Group eben durch eine gute Unternehmensarchitektur ermöglicht [TOG05]. Ihre Aufgabe ist die Analyse der fundamentalen Struktur von Systemen, Organisationen und deren Komponenten, um eine Strategie und die notwendigen Schritte zu ihrer Implementierung zu definieren. Da die Unternehmensarchitektur unternehmensweite architektonische Aspekte fokussiert, wird sie von Buhl et al. [BH04] als eine Makroarchitektur bezeichnet. Sie grenzt sich somit eindeutig von den Mikroarchitekturen ab, die sich auf Architekturen konkreter Anwendungen beziehen, muss jedoch gleichzeitig mit ihnen kompatibel sein.

Der Nutzen der Unternehmensarchitektur hat unterschiedliche Facetten:

- Vermeidung von Fehlentscheidungen in der IT und dem Geschäft sowie die frühzeitige Aufdeckung von Synergiepotenzialen [Sc03].
- Gestaltung effizienter IT-Prozesse bei gleichzeitiger Kostenreduzierung, z.B. Kosten des Softwaresupports [Sc03].
- Erhöhung des Return of Investment (ROI), z.B. durch Standardisierung der eingesetzten Technologien [Sc03].
- Senkung von Risiken, z.B. bei IT-Investitionen [TOG05, HVB03b].
- Steigerung des Unternehmen Marktwertes durch strategische Differenzierung [Sc03].

Zusätzlich fördert die Gestaltung der Unternehmensarchitektur die Transparenz der IT-Infrastruktur, Anwendungssysteme und Geschäftsprozesse. Diese Transparenz ist beispielsweise für einen Abschlussprüfer zwingende Vorraussetzung, um die Abgrenzung des Prüfungsumfangs sowie weitere prüferische Vorgehensweisen festlegen zu können [BV03].

Nach Gartner [La05a] darf eine Architektur jedoch nicht „as a goal unto itself“ gesehen werden. Die Architektur selbst liefert nicht direkt obigen Nutzen, sondern stellt lediglich den Rahmen bereit, mit der Technologielösungen näher an den Zielen des Geschäfts ausgerichtet werden können. Ross bezeichnet den Nutzen der Unternehmensarchitektur wie folgt: „The paybak for enterprise IT architecture efforts is strategic alignment between IT and the business. (…) Ultimately, enterprise architecture leads to „happy surprises“.“ [Ro03].

Die Schwierigkeit des Prozesses der Unternehmensarchitektur liegt in dem Nachweis dessen Nutzens und damit in der Motivierung einer übergreifenden Optimierung. Die Unternehmensarchitektur-Maßnahmen wirken nur sehr langfristig bei gleichzeitig schwierig zu messender Qualität. Der Nutzen der Architektur kann nur schwer durch Zahlen belegt werden, da dessen Effekte meist über das gesamte Unternehmen verteilt auftreten und nur selten konzentriert spürbar sind. Aus diesen Gründen ist der Rückhalt von der Top Management Ebene des Unternehmens zwingend erforderlich, um die kontinuierliche Weiterentwicklung der Unternehmensarchitektur zu gewährleisten [La05a, HVB03b].

Die Gestaltung und Umsetzung der Unternehmensarchitektur erfolgt durch die Architekten, die die Interessen der Stakeholder behandeln, indem sie dessen Bedürfnisse identifizieren und entsprechende Sichten der Architektur erstellen [TOG05]. Architekten bringen das Geschäft und die Technik durch enge und guter Kooperation mit vielen Einheiten zusammen. Ihre Kompetenz liegt in dem Verständnis von Geschäftsstrategien, -Anforderungen und den Technologien der IT. Sie sprechen und übersetzten die Sprache von Managern, Fachexperten und Technikern, um die Brücke zwischen Geschäft und IT zu schlagen. Die Herausforderung in der Arbeit der Architekten liegt dabei in dem Aufbau eines Vertrauensverhältnisses bei gleichzeitiger Steuerung und Kontrolle der Architektur [HVB03b].

Zusammenfassend lässt sich festhalten, dass eine Unternehmensarchitektur, die erfolgreich definiert und gelebt werden soll, mit der Geschäftsstrategie beginnen und bis zu der das Geschäft unterstützenden IT bearbeitet werden muss. Es ist wichtig, in kleinen und iterativen Schritten vorzugehen, um die Architektur stückweise im Unternehmen zu verankern. Hierzu bedarf es geeigneter Prozesse, Organisationsformen, Meilensteinen und einer streng einzuhaltenden Zeitplanung. Die Entwicklung und Umsetzung einer Unternehmensarchitektur erfordert harte Arbeit. Die Gefahr liegt dabei in einer anfänglichen Euphorie die über die Zeit verfliegt, so dass die Unternehmensarchitektur ein im Sand verlaufendes Projekt bleibt [La05a, HVB03b].

Phasen bei der Entwicklung einer Unternehmensarchitektur

Im vorangehenden Abschnitt wurde deutlich, dass die Erstellung, Weiterentwicklung und das Ausleben einer geeigneten Unternehmensarchitektur ein schwieriger, komplexer und nie endender Prozess ist. Bereits von Begin an muss dieser Prozess an den Kerngeschäftsprozessen des Unternehmens ausgerichtet werden.

Will ein Unternehmen den Architekturprozess anstoßen, stellt sich jedoch die Frage nach dem „wie". Ross hat für dieses Problem ein Modell mit vier Phasen entwickelt, dass zum einen die Klassifizierung der Ist-Unternehmensarchitektur ermöglicht und zugleich Anforderungen und Maßnahmen für die Verbesserung der Architektur beschreibt. Ein Unternehmen wird mit Hilfe des Modells entsprechend seiner Ist-Architektur einer Phase zugeteilt. Soll die Unternehmensarchitektur qualitativer werden, so muss das Unternehmen die nächste höhere Phase des Modells anstreben. Die vier architektonischen Phasen von Ross sind wie folgt definiert [Ro03]:

1. *Application silo architecture stage*: Befindet sich ein Unternehmen in der ersten Phase, so besteht die Unternehmensarchitektur aus mehreren Architekturen individueller Anwendungssysteme. Vorteil dieser Stufe ist die lokale Optimierung, da Anwendungssysteme entsprechend speziellen Anforderungen implementiert werden können. Des Weiteren existieren in der Entwicklung von Anwendungssystemen keine Einschränkungen, die Innovationen entgegen wirken könnten. Kosten und Budgets sind einfach abzuschätzen bzw. einzuteilen und Ergebnisse leicht messbar. Zu den Nachteilen dieser Stufe zählt dahingegen, dass Anwendungssysteme nur für den Spezialfall anwendbar sind, für den sie entwickelt wurden. Aufgrund der individuellen Architekturen sind Daten sehr stark an die Anwendungssysteme gebunden, das Management bei einer großen Anzahl von Anwendungssystemen einschließlich deren Integration fällt schwer.
2. *Standardized technology architecture stage*: Die Architektur eines Unternehmen der zweiten Phase erzielt ihren Wirkungsrad durch unternehmensweite Technologiestandardisierung und Zentralisierung. Die Vorteile bestehen darin, dass die IT sehr effizient wird und Kosten eingespart werden können. Darüber hinaus wird die Wartbarkeit der IT erleichtert und Ausfallsicherheit bzw. die allgemeine Sicherheit der Anwendungssysteme und Infrastrukturen erhöht. Ein großer Nachteil dieser Phase ist jedoch das Problem der Anwendungssystem-spezifischen Daten sowie die Eindämmung von Innovationen aufgrund der Technologiestandardisierung.
3. *Rationalized data architecture stage*: Unternehmen in der dritten Phase verfolgen neben der Technologiestandardisierung auch die Durchsetzung unternehmensweiter IT-Standards für Daten und Prozesse. Das Hauptaugenmerk dieser Phase liegt nicht mehr auf den Anwendungssystemen sondern auf dem Datenmanagement und dem Infrastrukturaufbau, indem Kerngeschäftsprozesse identifiziert und zentrale Speicher für Daten von Kernaktivitäten erstellt werden. Klarer Vorteil liegt in der Prozessoptimierung und der Gewichtung von Prozessen als Teil der Firmendefinition. Die dritte Phase ist jedoch mit hohen Risiken verbunden, da die Trennung der Daten von den Anwendungssystemen sehr schwierig ist und meistens eine große Umstrukturierung des Unternehmens erforderlich ist.
4. *Modular architecture stage*: In der vierten Phase baut die Architektur auf unternehmensweite, globale Standards mit lose gekoppelten IT-Komponenten auf. Globale Standards bei gleichzeitiger lokaler Differenzierung sollen in dieser Phase ermöglicht werden. Wesentlicher Vorteil dieser Stufe ist die Erstellung von widerverwendbaren Modulen. Geschäftsbereiche erhalten eine bessere Einsicht in ihre lokalen Prozesse, so dass die modulare Architektur an strategischer Mobilität gewinnt. Unternehmen in der vierten Phase laufen jedoch Gefahr, mit den Nachteilen der *application silo architecture stage* konfrontiert zu werden, falls Module ohne die Festlegung konkreter Einsatzszenarien und zugehöriger Kerndaten entwickelt werden.

Befindet sich ein Unternehmen, bzw. verschiedene Teilbereiche eines großen Konzerns, in einer der vier Phasen, so muss es auf der einen Seite phasenspezifische Anforderungen bzgl. ihrer Unternehmensarchitektur erfüllen, auf der anderen Seite kann es von der erreichten Unternehmensarchitekturqualität profitieren und Erfahrungen im Umgang mit der Architektur sammeln. Ein Übergang in die darüber liegende Stufe erfordert bestimmte Veränderungen innerhalb des Unternehmens, wobei laut Ross nur dann eine erfolgreiche Unternehmensarchitektur aufgebaut werden kann, wenn sich das

Unternehmen an die strikte Abfolge der vier Phasen hält. Eine Phase komplett zu überspringen sei ohne Rückschlag nicht möglich [Ro03].

Neben den Anforderungen und dem Mehrwert einer erreichten Phase bzw. Architektur, ändert sich auch das Verhältnis von IT und Geschäft bzw. Unternehmensarchitektur und Geschäftsstrategie. In der ersten Phase wird beispielsweise die Geschäftsstrategie definiert, ohne dass der Input der IT berücksichtigt wird. Auf gleiche Weise können innerhalb der IT-Entscheidungen getroffen werden, ohne dabei die Geschäftsstrategie zu beachten. Dieses Verhältnis unterliegt in der zweiten Phase bereits einem Wandel. In ihr treffen sowohl die Geschäfts- als auch die IT-Manager ihre Entscheidung nach einer gegenseitigen, jedoch noch recht oberflächlichen Absprache. Diese Kommunikation zwischen Geschäft und IT wird in der dritten Phase ausgebaut, d.h. es findet ein Austausch der geplanten strategischen Absichten, den IT-Leistungen und der Zielarchitektur statt. In der vierten Phase wird die Kommunikation soweit ausgebaut, dass sich Geschäft und IT gemeinsam mit den strategischen Entscheidungen auseinandersetzen.

Das Stufenmodell von Ross wurde in Gesprächen mit Spezialisten aus der Architekturgruppe der HVB Systems diskutiert. Es wurde der Aussage von Ross zugestimmt, dass eine getrennte Bewertung des Unternehmens anhand seiner Teilbereiche notwendig ist. Ein Unternehmen, insbesondere eine Bank, könne aus mehreren Bereichen mit unterschiedlichen Geschäftsmodellen bestehen, deren Ziele und Eigenschaft nicht zwingend kohärent sein müssten. Dabei könnten bewusst Silos entstehen, da eine Integration einiger Teilbereiche eher nachteilig sein könnte. Ein Zugriff auf Stammdaten würde beispielsweise absichtlich nicht für alle Bereiche standardisiert. Soll ein Unternehmen mit Hilfe des Stufenmodells klassifiziert werden, so bestehe der erste Schritt in der Kategorisierung des Unternehmens in mehrere Teilbereiche einschließlich des Geschäftsmodells. Bei einer Bank würde es dabei zu mehreren Teilbereichen kommen, als beispielsweise in einem Unternehmen im Telekommunikationssektor, da sie aufgrund ihrer umfangreichen Aufgaben, der komplexeren Anwendungslandschaft und den zahlreichen Schnittstellen in etwa um den Faktor fünf komplexer sei.

Des Weiteren könnten unterschiedliche Interessen eines Firmenteilbereichs, die Ursache für unterschiedliche Architekturen sein. Nicht in jedem Bereich stehe der Wunsch einer vollständigen Integration von Technologien, Daten und Prozessen an oberster Stelle. Wirft beispielsweise ein Bereich so viel Gewinn ab, dass die Kosten der Systeme nicht ins Gewicht fallen, wird kein Bedarf an der Erreichung der vierten Stufe des Modells von Ross gesehen. Der Mehrwert, den die Integration schaffen würde, stände nicht im Verhältnis mit den Kosten und den Risiken die auf dem Weg zur vierten Stufe anfielen. Erst im Falle nichtprofitabler Ergebnisse eines Bereiches würde der Fokus auf die Optimierung bzw. Standardisierung der Architektur gelenkt.

Es wurde festgestellt, dass bei der harten Trennung der vier Stufen und dem kontinuierlichen Übergang von einer Stufe in die nächste, das Problem darin bestehe, die Architektur der HVB Systems, bzw. deren Teilbereiche, eindeutig einer Stufe zuzuteilen. Werden die Stufen eingehalten, so befände sich die Bank in der zweiten Stufe. Dennoch bestünden Aktivitäten und Optimierungen der Bank, die sich im Bereich der Prozessstandardisierung (Stufe 3) und dem Aufbau widerverwendbarer Module (Stufe 4) abspielen. Die Standardisierung der Kerngeschäftsprozesse werden beispielsweise mit Hilfe des Werkzeugs PASS[8] umgesetzt. Aktivitäten im Bereich der serviceorientierten Architektur (SOA) und der Enterprise Application Integration (EAI) würden die Entwicklung widerverwendbarer Module fördern. Da jedoch die Bank auf Grund des schlechten Kosten-Nutzenverhältnisses, der Unternehmensstruktur und dem unterschiedlichen Umgang mit den Firmendaten (z.B. Risikomanagement vs. Effizienz) gegen die unternehmensweite Standardisierung von Daten sei, würde sie nach dem Stufenmodell von Ross niemals in die Stufe vier übergehen können.

Dennoch sind sich die Spezialisten darüber einig, dass die Zusammenarbeit von Geschäft und IT nicht optimal wäre. Erste Ansätze für eine gemeinsame Ausrichtung beider Parteinen müssten weiter ausgebaut werden, um beispielsweise die Abstimmung von Maßnahmen und Vorgehensweisen innerhalb von Projekten bereits von Beginn sowohl am Geschäft als auch an der IT auszurichten.

[8] PASS steht für „Prozesse, Architektur der IT, Strukturen, Standards“. Das Intranet Architekturportal stellt Prozesse mit Organisationen und IT-Systemen nach gleichartigen Grundmustern dar.

Zusammenfassend lässt sich aus dem vorangehenden Abschnitten festhalten, dass eine optimale Unternehmensarchitektur nicht an den Anforderungen der Vergangenheit, sondern an der Agilität der Organisation gemessen wird. Prozesse müssen so strukturiert sein, dass notwendige Änderungen schnell vollzogen werden können [CCI05]. Bei der Entwicklung der Unternehmensarchitektur sind dabei ein Verständnis relevanter Objekte verschiedenster Domänen des Unternehmens, sowie deren Zusammenspiel unverzichtbar. Die Einschätzung der Unternehmensarchitektur, wie sie beispielsweise durch das Stufenmodell von Ross durchgenommen werden kann, kann dabei als hilfreiche Orientierung für weitere Aktivitäten dienen. Liegt darüber hinaus ein integratives Modell für die benötigten Informationen vor, lassen sich benötigte Sichten entsprechend der Interessen von Stakeholdern generieren, die bei Entscheidungsfindungen als hilfreiches Werkzeug zur Verfügung stehen können. Durch die Entwicklung eines Modells für das System Unternehmen, kann eine Komplexitätsreduzierung[9] des selbigen erzielt werden [Es02].

Welche Werkzeuge bei dem Prozess der Unternehmensarchitektur unterstützend eingesetzt werden können, soll in der vorliegenden Arbeit am Beispiel der Softwarekartographie und einem Informationsmodell für das IT-Management erläutert werden. Der anschließende Abschnitt gibt eine Einführung in Softwarekarten, die in Kapitel 4 durch die Analyse bestehender Diagramme und Karten vertieft werden. Informationsmodelle für das IT-Management, die als Grundlage für Softwarekarten dienen können, werden in Kapitel 5 zunächst allgemein und anschließend mit Hilfe des in dieser Arbeit entwickelten Informationsmodells eingehend erläutert.

2.3 Softwarekartographie

Wie in den vorangehenden Abschnitten erläutert wurde, betrachtet das Architekturmanagement in Unternehmen verschiedenste Aspekte einer Unternehmung, die sich von den Strategien und Zielen über die Geschäftsprozesse und betrieblichen Anwendungssysteme bis hin zu Infrastrukturkomponenten erstrecken. Das Management der Unternehmensarchitektur soll dabei die Ausrichtung der IT an dem Geschäft des Unternehmens ermöglichen, indem die Kern-Geschäfts- und IT-Strategien sowie deren Einfluss auf die Geschäftsprozesse expliziert werden [LMW05c]. Dieser Prozess hat unmittelbaren Einfluss auf die Anwendungslandschaft des Unternehmens, der mit Hilfe der in Abbildung 3 veranschaulichten Betrachtungsebenen konkretisiert werden kann.

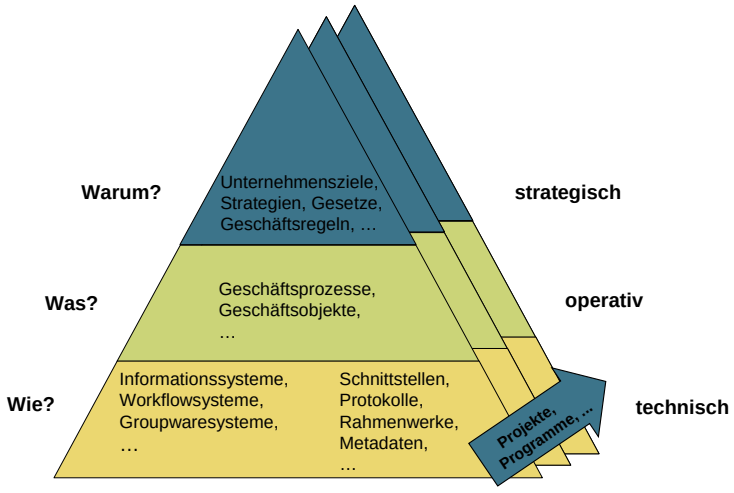


Abbildung 3: Betrachtungsebenen aus [LMW05c]

Die oberste Ebene der Unternehmensarchitektur, die Frage nach dem „Warum?“, umfasst die strategischen Ziele eines Unternehmens. IT-Strategien oder beispielsweise neue gesetzliche Regelungen können dabei Veränderungen der Geschäftsprozesse fordern und somit auf die Anwendungsland-

[9] Komplexitätsreduzierung ist die stärkste Möglichkeit von Vereinfachung, die es erlaubt, dieselbe Entscheidung zu finden, wie sie sich aus einem umfangreicheren Modell höherer Komplexität ergibt [Es02].

schaft einwirken. Beispielsweise setzt die Anforderung nach der Flexibilität eines Unternehmens Konzepte für die leichte Integration neuer Anwendungssysteme in die Anwendungslandschaft voraus.

Auf der zweiten Ebene („Was?“), den operativen Geschäftsprozessen und Geschäftsobjekten, besitzen Änderungen und Neuerungen direkte Auswirkungen auf die unterstützenden Anwendungssysteme. Anwendungssysteme müssen eventuell den neuen Anforderungen angepasst, abgelöst oder sogar vollständig neu entwickelt werden.

„Wie?“ Geschäftsprozesse und -Objekte implementiert und durch Anwendungssysteme unterstützt werden, wird auf der untersten Ebene Rechnung getragen. Der Fokus liegt hier auf den Technologien, Softwarearchitekturen und Schnittstellen der Anwendungssysteme.

Sollen diese drei Ebenen geeignet gesteuert werden, bedarf es Methoden und Modelle zur Beschreibung, Bewertung und Gestaltung von Anwendungslandschaften: die Softwarekartographie. Softwarekarten, als graphische Repräsentation der gesamten Anwendungslandschaft oder Ausschnitte selbiger, sollen die Analyse und die Gestaltung der Anwendungslandschaft unter Rückgriff auf Erkenntnisse und Methoden der Kartographie ermöglichen. Die Softwarekartographie liefert auf diese Weise einen Beitrag zum Architekturmanagement [LMW05c].

Die Kartographie ist ein Fachgebiet, das sich mit dem Sammeln, Verarbeiten, Speichern und Auswerten raumbezogener Informationen, sowie deren Veranschaulichung durch kartographische Darstellungen befasst [HGM02]. Sie stellt eine außerordentlich effiziente Möglichkeit dar, Ideen, Gedanken, Formen und Beziehungen auszudrücken, zu analysieren und zu manipulieren [Ro95]. Dazu liefert die Kartographie vor allem ein Zeichensystem, welches Merkmale und Regeln für alle graphischen Darstellungen beinhaltet [HGM02]. Dieses Zeichensystem soll in der Softwarekartographie verwendet werden, damit Softwarekarten als Kommunikationsmittel dienen, Zusammenhänge zwischen relevanten Aspekten hervorheben und spezielle Fragestellungen, wie beispielsweise das Erkennen von Redundanzen, die Koordination von Projekten oder das Identifizieren von Abhängigkeiten beantworten.

Softwarekarten können aus einem Kartengrund und mehreren darauf aufgetragenen Schichten bestehen, besitzen jedoch im Gegensatz zu Karten in der Kartographie keine geographische Verortung. Durch die Auswahl bestimmter Aspekte (s.u.) als Dimension, die zur Bildung des Kartengrundes (zur Verortung) zum Einsatz kommen, besitzt die Softwarekartographie im Gegensatz zur Kartographie zusätzlichen Freiraum bei der Gestaltung der Karte. Darüber hinaus können die angezeigten Informationen beispielsweise durch das Ein-/Ausblenden von Schichten gefiltert bzw. die Informationsdichte durch Zoom-In/Out variiert werden [LMW05a].

Nach Lankes et al. [LMW05a] kommen analog zur thematischen Kartographie, bei der auf topographischen Kartengründen thematische Inhalte visualisiert werden (z.B. die Bevölkerungsdichte), bei der Analyse von Anwendungslandschaften verschiedene Sichten zum Einsatz. Die Anwendungszwecke der einzelnen Sichten variieren und stellen je nach Anforderung unterschiedliche Aspekte bzw. Kennzahlen dar, die aus den Informationen der drei Betrachtungsebenen stammen. Welche Sichten bereits in der HVB Systems Verwendung finden, wird in den Kapitel 4 und Kapitel 6 erläutert.

Aspekte, die für eine Analyse der Anwendungslandschaft relevant sind und auf Softwarekarten visualisiert werden sollen, können nach Lankes et al. [LMW05a] in fünf Klassen eingeteilt werden:

1. *Planerische Aspekte*: Betrachtung der Evolution einer Anwendungslandschaft: Ist-, Plan- und Soll-Anwendungslandschaften. Ist-Anwendungslandschaften erfassen den Status quo. Plan-Anwendungslandschaften führen zu einer Anwendungslandschaft in der Zukunft und wurden bereits durch geplante Projekte und zur Verfügung gestelltes Budget konkretisiert. Soll-Anwendunglandschaften stellen schließlich zukunftsgerichtete Betrachtungen dar, die jedoch nicht durch geplante Projekte entstehen, sondern strategische Ziele der Anwendungslandschaften aufnehmen.
2. *Wirtschaftliche Aspekte*: Betrachtung der Kosten, die im Lebenszyklus eines Anwendungssystems entstehen.
3. *Fachliche Aspekte*: Prozesse, Organisationseinheiten, Geschäftsobjekte sowie Anzahl von Nutzern.
4. *Technische Aspekte*: Beispielsweise Implementierungssprachen, Verbindungen über Schnittstellen, genutzte Middleware oder Softwarearchitekturen von Anwendungssystemen.

5. *Operative Aspekte*: Fokus auf den Betrieb von Anwendungssystemen und die verbundenen Ereignisse.

Die Aspekte bieten sozusagen die Grundlage für den Inhalt einer Softwarekarte. In einem nächsten Schritt stellt sich die Frage nach der graphischen Kartengestaltung einschließlich ihres Aufbaus. Die Analyse des Forschungsprojektes Softwarekartographie ergab sich verschiedene Kartentypen, deren Zusammenhänge durch Abbildung 4 veranschaulicht werden.

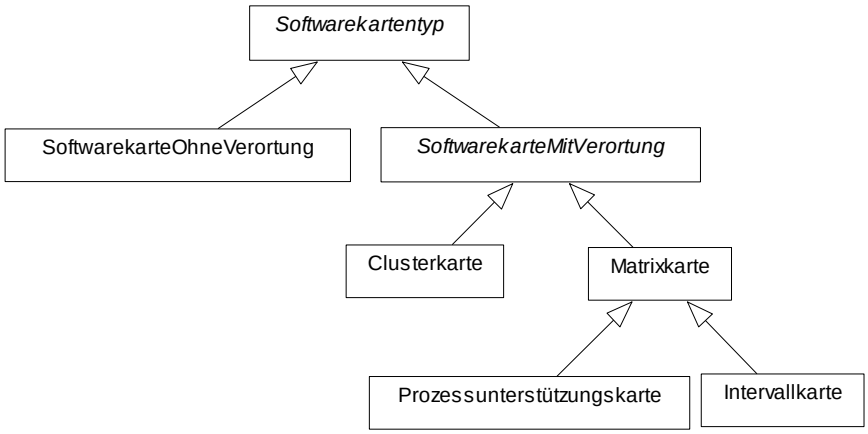


Abbildung 4: Softwarekartentypen[10]

Das Hauptentscheidungskriterium für den Kartentyp liefert die Verortung der Kartenelemente. Besitzt diese keine festgelegte Bedeutung, so spricht man von einer *Softwarekarte ohne Kartengrund*. Dieser Kartentyp kommt meistens zum Einsatz, wenn eine Visualisierung der Anwendungslandschaft oder deren Ausschnitte speziell auf eine Problemstellung hin generiert werden soll. Verortungsregeln sind beispielsweise die Minimierung von Überschneidungen der Verbindungslinien, gleichmäßige Verteilung der Kartenelemente über die Karte, Verbindungslinien ähnlicher Länge oder gleiche Ausrichtung gerichteter Verbindungen.

Sind dahingegen für die Verortung von Kartenelementen bestimmte Kriterien festgelegt, handelt es sich um *Softwarekarten mit Kartengrund zur Verortung*. Erscheint ein bestimmtes Element an einer anderen Stelle der Karte, ändert sich die von ihm transportierte Botschaft. Softwarekarten dieses Typs lassen sich weiter klassifizieren in die Matrix- und die Clusterkarten.

Eine *Matrixkarte* richtet ihre Elemente tabellenartig an einer vertikalen und einer horizontalen Achse (bzw. Spalte und Zeile oder x- und y-Achse) aus. Die Visualisierung der HVB Systems Technologiesets (siehe Anhang A) ist beispielsweise eine Matrixkarte, die Produkte auf der x-Achse anhand ihrer Lebenszyklen-Stati und auf der y-Achse aufgrund ihrer Produktklassifizierung verortet.

Im Forschungsprojekt Softwarekartographie wurden darüber hinaus zwei, bei den Projektpartnern häufig auftretende, Matrixkarten identifiziert und gemäß den an den Achsen angebrachten Informationen als eigener Kartentyp spezifiziert. Es handelt sich um die Intervall- und die Prozessunterstützungskarte.

Die *Intervallkarte* (siehe Abbildung 5) setzt als charakterisierende Notationstechnik Balken ein, deren Länge die Dauer von Vorgängen (z.B. Projekten) oder von Lebenszyklusphasen (z.B. eines Anwendungssystems) repräsentiert. Bei der Intervallkarte tritt der Aspekt Zeit als x-Achsen Dimension zur Verortung in den Mittelpunkt, die y-Achse kann durch den Kartengestalter frei gewählt werden. Diese Diagrammart ist an die Gantt-Darstellung angelehnt.

[10] Vgl. Visualisierungsmodell Abschnitt 3.1.4.

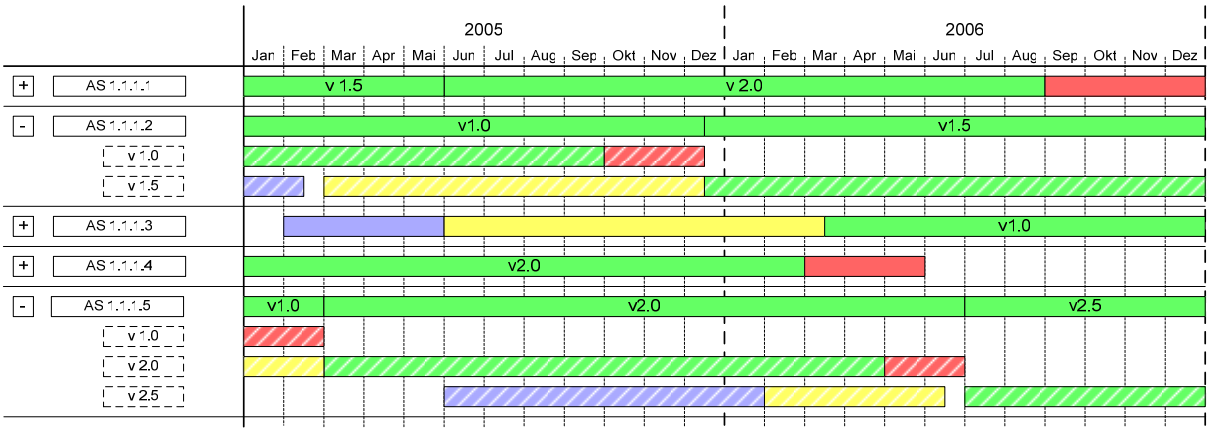


Abbildung 5: Intervallkarte nach [LMW05c]

Die *Prozessunterstützungskarte* verortet dahingegen Anwendungssysteme anhand der von ihnen unterstützten (betrieblichen) Prozesse (siehe Abbildung 6). Im Rahmen der Prozessorientierung und Unternehmensarchitekturen erhalten organisationseinheitsübergreifende Prozesse innerhalb von Unternehmen eine größere Bedeutung. Obwohl zwischen den einzelnen Prozessschritten Schnittstellen existieren, besteht ein Ziel der Prozessorientierung (insb. auf den höheren Ebenen) darin, den kontinuierlichen Verlauf der Wertschöpfung im Prozess zu betonen. Die Unternehmensarchitektur gibt zusätzlich eine Gesamtsicht unter Berücksichtigung verschiedener Perspektiven auf das Unternehmen.

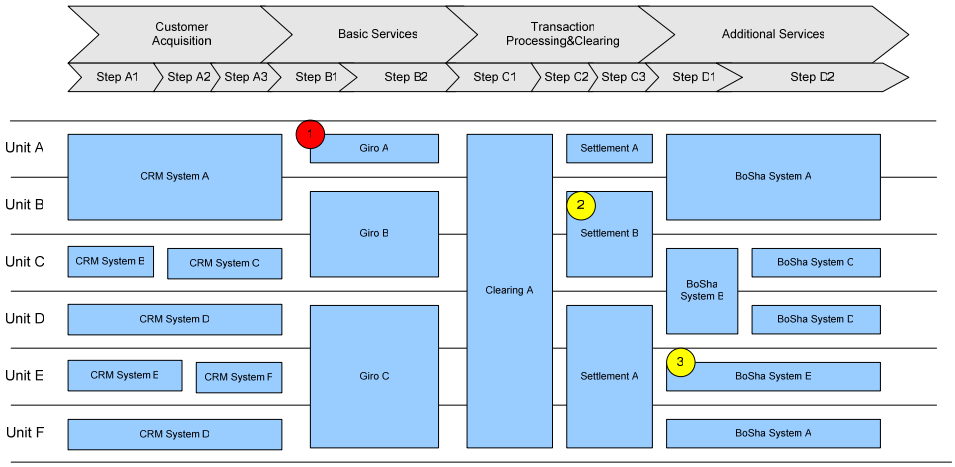


Abbildung 6: Prozessunterstützungskarte nach [LMW05c]

Im Gegensatz zur Matrixkarte handelt es sich bei der *Clusterkarte* um eine Softwarekarte mit Kartengrund zu Verortung, die das Problem der Verortung von Elementen löst, indem jedes Element in der logischen Einheit dargestellt wird, zu der es in Beziehung steht (siehe Abbildung 7). Damit stehen beim Erstellen einer derartigen Karte bereits grobe Regeln fest, wo z.B. ein bestimmtes Anwendungssystem darzustellen ist. Wenn zwischen den logischen Einheiten, die den Kartengrund bilden und den Anwendungssystemen eine n:m Beziehung besteht, existiert die Möglichkeit, dass ein System auf der Karte mehrmals erscheint. Zu beachten ist, dass dieser Kartentyp nicht spezifiziert, wie die logischen Einheiten auf der Karte platziert werden und wie sich die verschiedenen Elemente innerhalb der Darstellung einer logischen Einheit anordnen. Die relative Position der Cluster zueinander besitzt keine formale Bedeutung, es sei denn sie sind ineinander geschachtelt.

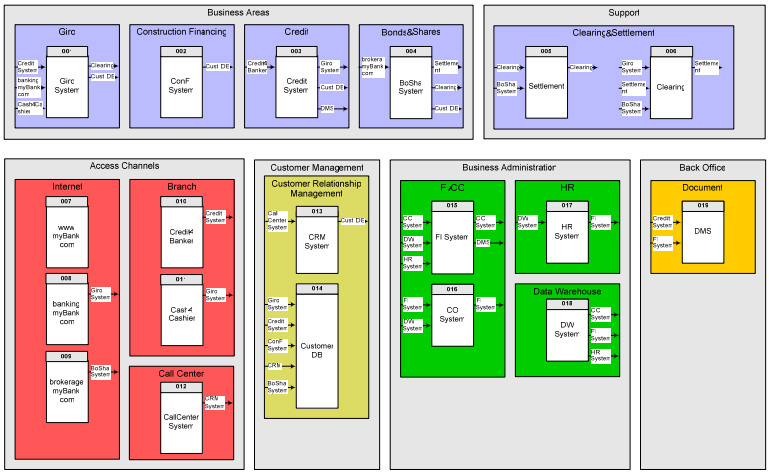


Abbildung 7: Clusterkarte nach [LMW05c]

Bereits in den Anfängen des Forschungsprojektes wurde deutlich, dass der Aufwand und die Kosten, die die Erstellung und Pflege von Softwarekarten verursachen, sehr hoch sind. Um diesen entgegenzuwirken, hat sich das Forschungsprojekt die Entwicklung eines Werkzeuges zur Softwarekartographie als Ziel gesetzt, dass die vollautomatisierte, regelbasierte Erstellung von Softwarekarten ermöglicht, indem es in eine bereits existierende Systemumgebung mit Repository integriert werden kann. Hierfür benötigte Konzepte sind ein einheitlicher Begriffsapparat zur Beschreibung von Anwendungslandschaften in Bezug auf das Unternehmensarchitektur-Management und der Aufbau von Modellen für die Komponenten der Softwarekartographie (Visualisierungsmodell) und den zu visualisierenden Informationen (Informationsmodell). Der Hauptbeitrag, den die Softwarekartographie zum Architekturmanagement leistet, ist die Bereitstellung und die Methodik zur Dokumentation der Architektur von Anwendungslandschaften bzw. der Unternehmensarchitektur. Zu dieser Methodik zählt neben der Notation selbst auch die Orientierung an *best-practices* auf dem Gebiet der Architekturdokumentation. Der Einsatz unterschiedlicher *Views* zur Darstellung verschiedene Aspekte des betrachteten Systems, das Trennen von *Views* und deren Abstraktionen (den *Viewpoints*) sowie die Ausrichtung der Beschreibung an die Bedürfnisse von Stakeholdern sind einige der bewährten Verfahren [LMW05c].

Die zuvor genannten *best-practices* werden in Kapitel 4 detaillierter erläutert und anhand der Analyse bestehender Softwarekarten der HVB Systems mit Beispielen untermauert. Der Schwerpunkt der vorliegenden Arbeit liegt in der anschließenden Entwicklung eines Informationsmodells, das die auf den Softwarekarten zu visualisierenden Objekte und Aspekte der Unternehmensarchitektur miteinander in Beziehung setzt. Um das Verständnis und die Einordnung des Informationsmodells zu gewährleisten, soll zunächst das folgende Kapitel benötigte theoretische Grundlagen darlegen.

Kapitel 3

Theoretische Grundlagen

Bevor auf den zentralen Teil dieser Arbeit, bestehend aus der Kartenanalyse, sowie der Realisierung und Implementierung des Informationsmodells, eingegangen werden kann, sollen in diesem Kapitel die theoretischen Grundlagen für die Modellierung eingeführt werden. Die Frage: „Was ist unter einem Modell zu verstehen?“ und „Was ist ein Informationsmodell?“ werden einleitend im Zusammenhang des Begriffes der Metamodellierung erläutert (siehe 3.1.1 - 3.1.3). Im darauf folgenden Abschnitt werden weitere Klassifizierungen von Modelltypen, dem semantischen und dem symbolischen Modell vorgenommen (vgl. 3.1.4), die für das Verständnis, der in Abschnitt 4.3 und 5.2 entwickelten Modelle, dienen. Der Einführung in die Modellierungssprachen MOF[11] und UML[12] in Abschnitt 3.2 folgen abschließend die in dieser Arbeit verwendeten Design-Prinzipien (siehe 3.2.3).

3.1 Modell Begriffsdefinitionen

Der Begriff des Modells findet in dieser Arbeit häufig Verwendung. Aus diesem Grund sollen zunächst unterschiedliche Begriffsdefinitionen zum Modellbegriff angeführt werden.

3.1.1 Modell

Die Definition des Begriff *Modells* leitet sich durch die drei Hauptmerkmale des allgemeinen Modellbegriffs her (angelehnt an [St73]). Ein Modell ist stets ein Modell von etwas, d.h. von Abbildungen bzw. Repräsentationen natürlicher oder künstlicher Originale, die selbst wieder Modelle sein können. Diese Eigenschaft bezeichnet man als das *Abbildungsmerkmal*. Das zweite Kennzeichen ist das *Verkürzungsmerkmal*, da Modelle im Allgemeinen nicht alle Attribute des durch sie repräsentierten Originals erfassen, sondern nur solche, die den jeweiligen Modellerschaffern und/oder Modellbenutzern relevant scheinen. Dritte Charakteristik stellt das *pragmatische Merkmal* dar, was bedeutet, dass Modelle ihren Originalen nicht per se eindeutig zugeordnet sind. Sie erfüllen ihre Ersetzungsfunktion für bestimmte – erkennende und/oder handelnde, Modellbenutzende – Subjekte, innerhalb bestimmter Zeitintervalle und unter Einschränkung auf bestimmte gedankliche oder tatsächliche Operationen.

Das pragmatische Merkmal vereint zwei grundlegende Auffassungen, die in der Literatur für den Begriff des Modells eine entscheidende Rolle spielen. Wird ein Modell einerseits vom *abbildungsorientierten* Standpunkt aus gesehen, dient es dazu, die Realität homomorph, d.h. struktur- und verhaltensgetreu wiederzugeben, um komplexe Phänomene untersuchen und erklären zu können [LHM95]. Ist ein Modell andererseits immer durch die subjektive Interpretation der Realität geprägt, so handelt es sich um einen *konstruktionsorientierten* Modellbegriff. Neben der nüchternen Realitätsbetrachtung enthalten solche Modelle auch immer Fiktionen und haben somit Struktur gebenden Charakter [Sc94].

3.1.2 Metamodell

Für die Konstruktion von Modellen (auch Modellsystem genannt), werden nach Esser [Es02] Konstrukte, Strukturen und syntaktische Beziehungsregeln benötigt, die in Form von so genannten *Metamodellen* als Methodenwissen (Arbeitstechniken, Vorgehensweisen, Werkzeuge) gebündelt werden. Mit Hilfe von Metamodellen wird die Einhaltung der Konsistenz und Vollständigkeit des Modellsystems unterstützt.

[11] Meta Object Facility.

[12] Unified Modelling Language.

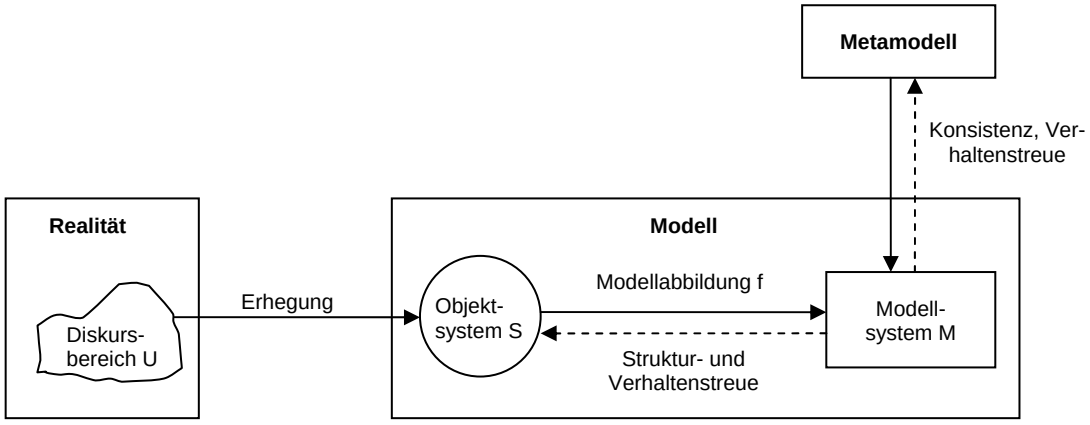


Abbildung 8: Struktur des Modellbegriffs aus [Es02]

Abbildung 8 veranschaulicht die Zusammenhänge von Realität, Modell und Metamodell. Ein Ausschnitt der Realität, ein Diskursbereich, wird subjektiv interpretierend erhoben, so dass ein *Objektsystem* durch Vereinfachungen der Bestandteile des Diskursbereiches entsteht. Objekte und Beziehungen des zu modellierenden Systems werden dabei zweckorientiert ausgewählt, benannt und im Objektsystem gebunden.

Unter Beachtung eines Modellzwecks, wird das Objektsystem in ein *Modellsystem* abgebildet. Für ein Objekt des Objektsystems (z.B. Inge Meier), wird eine Klasse im Modellsystem modelliert (z.B. Kunde). Die Abbildung von Objekt- in Modellsystem kann als die Modellabbildung bezeichnet werden. Das Modellsystem wird gegenüber dem Objektsystem abstrakter und weniger konkret. Die Syntax für die Erstellung des Modellsystems wird dabei im *Metamodell* festgelegt.

3.1.3 Informationsmodell

Modelle können aufgrund ihrer Thematiken weiter klassifiziert werden. Der für diese Arbeit entscheidende Modelltyp ist der des Informationsmodells. Dieser findet in unterschiedlichen Anwendungsgebieten Verwendung und wird in der Literatur häufig differierend definiert.

Teubners Definition lautet: "Informationsmodelle bilden nicht nur einzelne Informationssysteme ab, sondern das Unternehmen bzw. die Unternehmensorganisation und stellen damit einen Bezugsrahmen für die Beschreibung und organisatorische und softwaretechnische Integration der Informations- bzw. Anwendungssysteme dar. (...)" [Te99]. Dahingegen ist nach Becker, Schütte [BS96] ein Informationsmodell "das immaterielle Abbild des betrieblichen Objektsystems aus Sicht der in diesem verarbeiteten Informationen für Zwecke des Informationssystem- und Organisationsgestalters."

Den Definitionen ist gemeinsam, dass ein Informationsmodell einen Bezug zu einem betriebswirtschaftlichen System aufbaut. Die der Arbeit zugrunde liegende Definition lautet:

Ein *Informationsmodell* ist im Speziellen ein Modell, welches die Beziehungen zwischen Informationsobjekten, sowie deren Attribute darstellt. Ein Informationsobjekt ist dabei eine Abstraktion eines real existierenden Objektes, dass für das Modell relevante Informationen über das Objekt aggregiert. Das Informationsmodell arbeitet mit Begriffen wie "Informationssystem", "Prozess, "Projekt" etc. All diese Begriffe werden auf Entitätstypen im Informationsmodell abgebildet, die im objektorientierten Paradigma einer Klasse entsprechen. Das Informationssystem "FiBu" (Finanzbuchhaltung) ist ein Objekt und somit eine Instantiierung der Klasse Anwendungssystem (Notation: "FiBu:Anwendungssystem").

Informationsmodelle sind keine Metamodelle zur Prozessmodellierung, die auf Funktionen, Ereignisse, Daten etc. fokussieren. Ebenso wenig sind es reine Metamodelle zur Datenmodellierung, in denen es sich um Objekte und deren erzeugte Daten handelt, die in einer Datenbank abgelegt werden sollen z.B. Organisationseinheitsdaten, Personendaten, Abteilungsdaten. In einem Informationsmodell wird

nur dann auf Applikationsdaten eingegangen, wenn sie eine Rolle für die Anwendungssysteme und relevante Aspekte spielen.

Der Titel des in Kapitel 5.2 zu entwickelnde *Modell für das IT-Management* gibt darüber hinaus Hinweis auf die verbundenen Ziele des in dieser Arbeit zu erstellende Informationsmodells. Der Fokus liegt auf solchen Objekten, einschließlich ihren Attributen und gegenseitigen Beziehungen, die für die Unternehmensarchitektur und deren Management relevant sind. Es besitzt einen gesamthaften Ansatz der Beschreibung und Umsetzung von Prozessen, IT-Architekturen, Strukturen und Standards, sowie Themen der Unternehmens- und IT-Strategie. Der Schwerpunkt der Betrachtung liegt dabei auf dem statischen Aufbau, d.h. den Objekten und deren Beziehungen. Das Verhalten des Systems und seiner Komponenten über den Zeitverlauf ist nicht Aufgabe des Informationsmodells.

Das Informationsmodell für das IT-Management hat überwiegend abbildungsorientierten Charakter. Da jedoch einige Objekte im Hinblick auf eine wünschenswerte Zukunftssituation benannt und eingeführt wurden, ist es in Teilen auch konstruktionsorientiert.

3.1.4 Semantisches und symbolisches Modell

Der Begriff des Informationsmodells kann mit Hilfe weiterer Modelltypen in den Kontext der Softwarekartographie eingeordnet werden, die im Folgenden beschrieben werden.

Sollen Aufwand und Kosten reduziert werden, die bei der manuellen Erstellung und Pflege von Softwarekarten entstehen, so bedarf es einer regelbasierten und automatisierten Erstellung von Softwarekarten. Der erste Schritt dieses Automatismus liegt in der klaren Trennung von zu visualisierenden Informationen bzw. Objekten oder Aspekten der Unternehmensarchitektur und den Konzepten der Softwarekartographie, d.h. die Definition eines semantischen und eines symbolischen Modells [LMW05c], wie sie bereits in den Arbeiten von Torre et al. [To04] vorgenommen wird (siehe Abbildung 9).

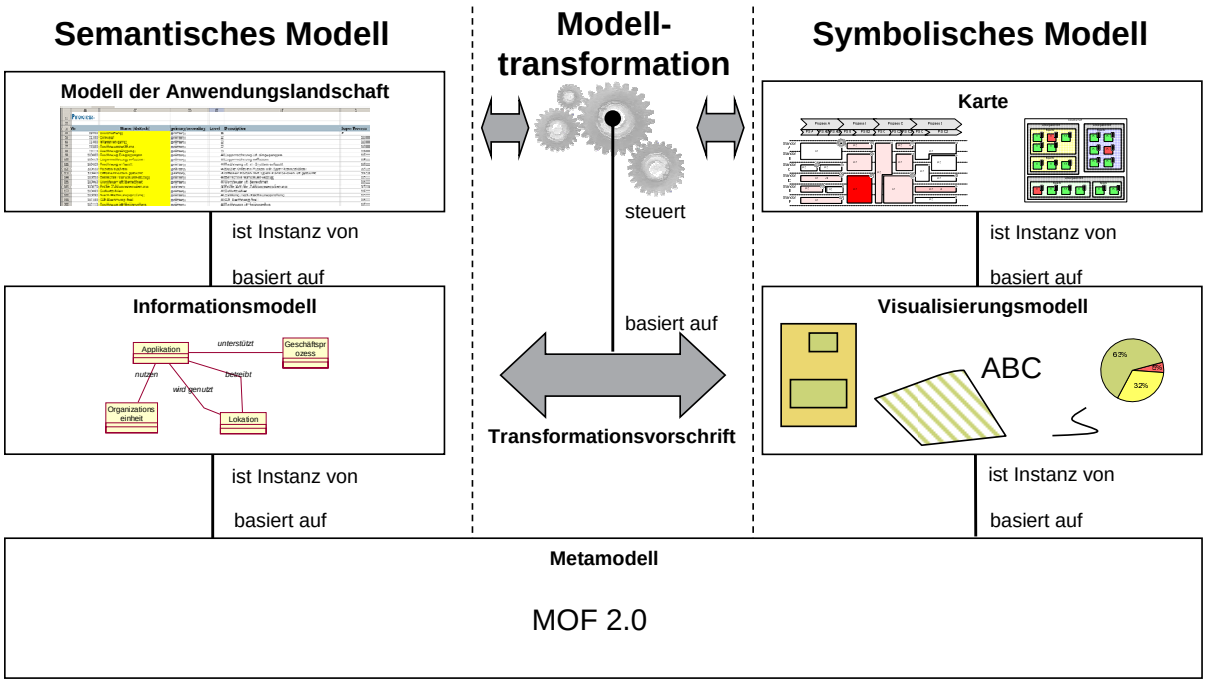


Abbildung 9: Zusammenhang des semantischen und symbolischen Modells aus [LMW05c]

Im semantischen Modell werden die Objekte bzw. Informationen über die Unternehmensarchitektur und die Anwendungslandschaft definiert und spezifiziert. In ihm sind nur die Inhalte der Informationsobjekte selbst relevant, nicht ihre Darstellungsform. Das symbolische Modell stellt dahingegen die Informationsobjekte des semantischen Modells über ein geeignetes Visualisierungsmodell z.B. Softwarekarten, Diagramme oder tabellarische Reports, dar. Durch das symbolische Modell werden die Informationsobjekte des semantischen Modells erfahrbar und kommunizierbar.

3.1.4.1 Semantisches Modell

Basisbestandteile des semantischen Modells sind die Informationsobjekte, d.h. Abstraktionen der real existierenden Objekte, die für das Modell relevante Informationen über Objekte aggregieren. Um von der Objektebene auf eine Klassenebene bzw. Entitätstyp zu abstrahieren, wird ein so genanntes Informationsmodells entwickelt, dass die relevanten Informationsobjekte, dessen Attribute und Beziehungen in einem geschlossenen Modell (z.B. UML Klassenmodell der OMG) abbildet. Die Entwicklung eines solchen Informationsmodells für die Unternehmensarchitektur der HVB Systems ist die zentrale Aufgabe dieser Arbeit und Bestandteil des Kapitel 5.

Werden die Inhalte des Informationsmodells um eine weitere Ebene abstrahiert, werden die Sprachkonstrukte in Form eines Metamodell definiert. Für ein Informationsmodell in Form eines UML Klassenmodells könnte beispielsweise das Metamodell eine MOF Beschreibung sein. MOF steht für Meta Object Facility [OMG03b], einer Architektur für Metamodelle. Der MOF Standard gibt Regeln, Randbedingungen und Elemente zur Erstellung von MOF-konformen Metamodellen vor. Eine MOF Beschreibung für das UML Klassenmodell setzt beispielsweise die Begriffe Klasse, Assoziation und Attribute miteinander in Beziehung.

3.1.4.2 Symbolisches Modell

Soll das semantische Modell kommunizierbar und erfahrbar gemacht werden, so bedarf es eines symbolischen Modells. Die unterste Ebene bilden hier analog zu den Informationsobjekten, die Visualisierungsobjekte wie beispielsweise unterschiedliche Farbtöne, Schriften oder Softwarekartentypen. In welchem Zusammenhang diese Visualisierungsobjekte miteinander stehen und durch welche Attribute sie klassifiziert werden, ist Teil des Visualisierungsmodells. Es gibt für Darstellungen vor, wie sie organisiert sind, d.h. welche Darstellungen überhaupt möglich sind. Diese Beschreibung soll dabei in einer Art und Weise erfolgen, die einerseits eindeutig angibt, was auf der Darstellung gezeigt werden soll und andererseits auch eindeutig erkennen lässt, welche Freiräume für die Erstellung der Darstellungen geboten werden. Ein erster konzeptueller Entwurf für ein Visualisierungsmodell der Softwarekartographie ist Teil dieser Arbeit (siehe Abschnitt 4.3).

Eine Abstraktion auf die Ebene der Metamodelle muss auch für das symbolische Modell entwickelt werden, wenn es Konformität und Kompatibilität verschiedener Visualisierungsmodelle erzielen möchte. Eine Metaebene, wie sie bereits im semantischen Modell eingeführt wurde, ist somit Bestandteil des symbolischen Modells.

3.1.4.3 Transformation

Um dem Ziel einer regelbasierten und automatisierten Erstellung von Softwarekarten gerecht zu werden, bedarf es einer Verbindung des semantischen und des symbolischen Modells, so dass aus den über die Unternehmensarchitektur bzw. Anwendungslandschaft verfügbaren Informationen eine geeignete Visualisierung erfolgen kann.

Diese Transformation soll weitestgehend automatisiert laufen, indem geeignete Transformationsregeln für die Art und Weise der Visualisierung einschließlich der Verortung von Informationsobjekten des semantischen Modells aufgestellt werden können. Es muss genau spezifiziert werden, welche Elemente auf der Seite des semantischen Modells auf welche Elemente auf der Seite des symbolischen Modells abgebildet werden. Dabei muss dem Modellierer neben den Transformationsregeln eine gewisse Gestaltungsfreiheit erhalten bleiben, die das nachträgliche Anpassen und Optimieren der Softwarekarte ermöglicht. Händische Veränderungen des Modells, durch die eine Informationsverfälschung vorgenommen wird, sind jedoch zu unterbinden oder müssen zu entsprechenden Veränderungen der Informationsobjekte führen.

Das Forschungsprojekt Softwarekartographie arbeitet derzeit an Transformationsregeln und einem Visualisierungsmodell inkl. einem passenden Metamodell, um die Nutzung der „Bidirectional Object Oriented Transformation Language“ (BOTL), einer partiell bijektiven Abbildung zwischen den Modellen, zu ermöglichen [LMW05c].

3.2 Modellierungssprachen und ihre Konstrukte

Nachdem sich die ersten Abschnitte dieses Kapitels mit dem grundsätzlichen Begriff des Modells auseinandergesetzt haben, stellt sich die Frage nach der Art und Weise der Modellierung.

Wie bereits oben eingeführt, ist ein Modell eine Abbildung bzw. eine Repräsentation eines natürlichen oder künstlichen Originals. Zur Darstellung von Objekten und ihren Beziehungen wird dafür eine geeignete Modellierungssprache benötigt, deren Spracheignung durch die Auswahl einer dem Problem entsprechenden Modellierungssprache einschließlich relevanter Konstrukte innerhalb dieser Sprache, bestimmt wird [RS02].

3.2.1 Klassifizierung von Modellierungssprachen

Modellierungssprachen lassen sich nach den Kriterien textuell vs. grafisch und informal vs. formal klassifizieren [Ba00].

Eine *textuelle Beschreibung* legt die Anforderungen in Form von natürlich sprachlichen Texten fest. *Graphische Darstellungen* legen dahingegen die Anforderungen durch grafische Symbole und Linien zwischen den Symbolen fest, wobei Symbole und Linien meistens durch Namen oder textuelle Symbole bezeichnet werden. Auch eine grafische Darstellung kann durch eine Grammatik festgelegt werden, wenn die grafische Syntax den Aufbau der Grafik bestimmt.

Eine *formale Beschreibung* findet unter Verwendung formaler, textueller Sprache statt. Eine zugehörige Grammatik definiert dabei, welche Zeichenketten erlaubte Sätze der Sprache sind. Im Gegensatz dazu ist eine *informale Modellierungssprache* in ihrer Beschreibung an keine Regeln gebunden.

Zwischen den formalen und informalen Beschreibungen liegen die semiformalen oder formalisierten Sprachen. Diese sind stärker formalisiert als die Umgangssprache, aber nicht streng mathematisch wie eine formale Sprache mit ihrer Grammatik.

In den Bereich der *textuell-informalen* Sprache fällt somit die verbale Anforderungsbeschreibung, d.h. die Beschreibung durch Worte aus der Umgangssprache. Das klassische Beispiel ist hierfür der reine Text. Zu den *textuell-formalen Sprachen* zählen beispielsweise textuelle Petri-Netze, die reinen Petri-Netze gehören zu der *grafisch-formalen Sprache* [Re82].

Die Modellierungssprache der Modelle dieser Arbeit ist die semiformale UML. Um das Verständnis des Lesers zu unterstützen, werden die Modelle zusätzlich textuell beschrieben.

In der Literatur finden sich alternativ zur UML auch häufig Entity Relationship Diagramme, die ursprünglich zur Datenmodellierung entwickelt wurden [Sc01, Ba00]. Da die folgenden Modelle jedoch durch zahlreiche Attribute gekennzeichnet sind, wurde der objektorientierte Ansatz vorgezogen, welcher der Anforderung nach einer geeigneten Strukturierung, Übersichtlichkeit und Lesbarkeit gerecht wird [Ba98].

Im folgenden Abschnitt soll auf die in dieser Arbeit benötigten Konzepte der Modellierungssprache eingegangen werden, die sowohl auf der MOF (Meta-Object Facility) Core Specification 2.0 als auch der UML Infrastructur Specification 2.0 basieren.

3.2.2 Modellierung mit der MOF und UML

Die *Meta-Object Facility Spezification* (MOF) beschreibt eine abstrakte Sprache und ein Framework zu Verwaltung von plattformunabhängigen Metamodellen, wie beispielsweise die *Unified Modeling Language* (UML) [OMG03a, OMG03b]. Während MOF einfacherer, direkt zu implementierende Konstrukte zur Metamodellmodellierung benötigt, stellt die UML eine allgemeine grafische Notation zur Erstellung von Systemen dar. Sowohl MOF als auch UML stellen eine Sprachspezifikation (Metamodell) dar, mit der Benutzer ihre eigenen Modelle definieren können. Aus der Perspektive von MOF ist UML eine Spezifikation, die auf MOF als Sprachspezifikation basiert. MOF ist demzufolge das Metamodell der UML.

Abbildung 10 veranschaulicht die unterschiedlichen Ebenen der Modellierung anhand von MOF und UML (vgl. Abschnitt 3.1).

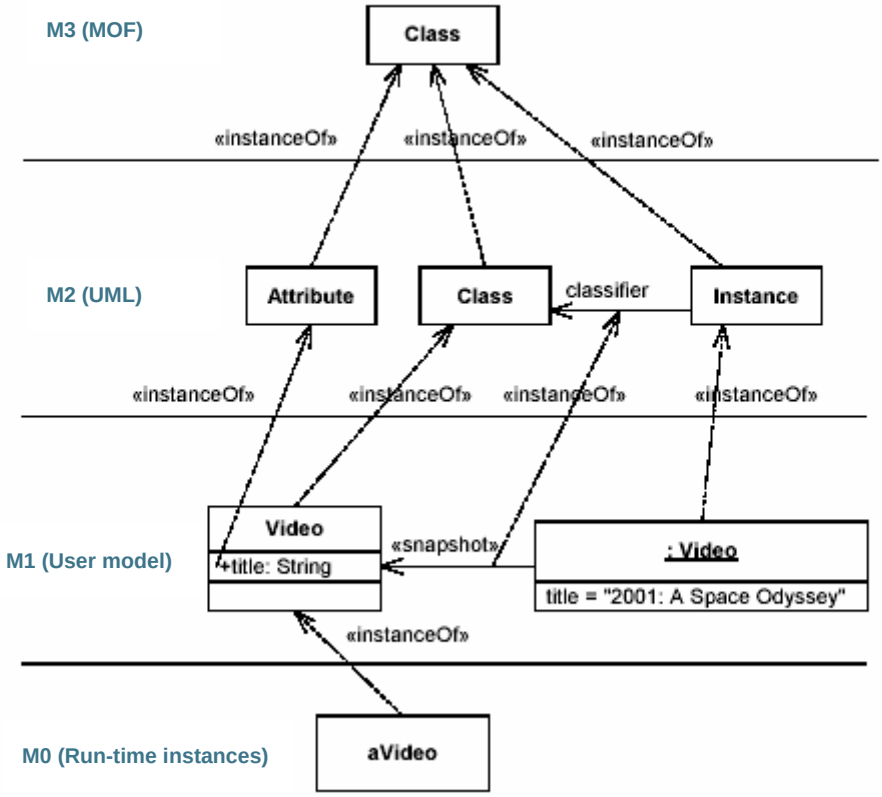


Abbildung 10: Metamodell-Hierarchie in Bezug auf MOF und UML nach [OMG03a]

Die Meta-Metamodellierungsebene (M3) hat die Aufgabe, eine Sprache (z.B. MOF) für die Spezifikation eines Metamodells zu definieren. Ein Meta-Metamodell ist häufig kompakter als die von ihm beschriebenen Metamodelle und teilt häufig die gleichen Design Philosophien und Semantiken wie die Metamodelle.

Ein Metamodell ist eine Instanz des Meta-Metamodells, d.h. jedes Element des Metamodells ist eine Instanz eines Elementes des Meta-Metamodells. Die Metamodellebene (M2) definiert dabei die Sprache für die Spezifikation von Modellen. UML M2 Modelle sind beispielsweise Instanzen von MOF Modellen und ermöglichen die Spezifikation von Klassen-, Use-Case[13]- oder z.B. Sequenzdiagrammen.

Die Instanz eines Metamodells ist ein Modell der Modellebene (M1), dessen Sprache semantische Domänen beschreibt. Ein Anwendermodell (engl. user model), z.B. ein konkretes UML Klassendiagramm, ist beispielsweise eine Instanz des UML Metamodells. Es enthält darüber hinaus sowohl Modellelemente (z.B. Video), als auch Abbilder von Instanzen (z.B. Video mit Titel „A Space Odyssey“).

Die unterste Ebene (M0) ist die Ebene der Laufzeitinstanzen (engl. Run-time Instances) des Modells, z.B. ein konkretes Video.

Neben dem Zusammenhang von MOF und UML als Meta-Metasprache der Metasprache, existiert eine weitere Abhängigkeit: die UML 2.0 Infrastructure [OMG03a] wird in der Definition der MOF 2.0 verwendet.

Folgend werden die Konstrukte und Konzepte von MOF bzw. UML aufgelistet, die in den Modellen dieser Arbeit eingesetzt wurden.

[13] Deutsch Anwendungsfall.

1. *Element*: Ein Element ist ein Bestandteil eines Modells. Es wird unterschieden zwischen benannten Elementen (engl. NamedElement) die einen Namen besitzen und den typisierten Elementen (engl. TypedElement), die benannte Elemente mit einem Typ repräsentieren.
2. *Typ* (engl. Type): Unter einem Typ versteht man ein benanntes Element, das als Typ für ein typisiertes Element verwendet wird.
3. *Datentyp* (engl DataTyp): Ein Datentyp ist eine abstrakte Superklasse für unterschiedliche Datentypen, wie beispielsweise den *primitiven Typen* (engl. PrimitiveType). Bei letzteren wird zwischen *Integer* (alle unendliche ganze Zahlen), *String* (Zeichenketten), *Boolean,* (Datentyp für logische Ausdrücke mit den zwei vordefinierten Typen true und false) und *UnlimitedNatural* Datentyp unterschieden. UnlimitedNatural Datentypen repräsentiert unendliche natürliche Werte und findet in den folgenden Modellen bei der Angabe der Kardinalitäten an den Assoziationsenden Verwendung.
4. *Paket* (engl. Package): Ein Packet wird zur Gruppierung von Elementen (Typen oder Pakete) verwendet und liefert gleichzeitig einen Namensraum für die enthaltenen Elemente [OMG03a]. Es kann mittels einem URI (Uniform Ressource Identifier) eindeutig identifiziert werden und Elemente anderer Pakete verwenden.
   Sollen Elemente eines Pakets in einem anderen Paket sichtbar sein, um sie dort mit weiteren Assoziationen zu versehen bzw. durch Vererbungen zu spezialisieren, so bezeichnet man die Beziehung zwischen den Paketen als *Import*. Sollen dahingegen Erweiterungen in Form neuer Metamodellierungseigenschaften vorgenommen werden, so handelt es sich um eine *Merge* (deutsch mischen) bzw. *Combine* (deutsch kombiniert). Die Beziehungsart *Merge* ermöglicht es, Elemente eines Pakets in einem anderen Paket mit neuen Eigenschaften zu versehen. Dabei handelt es sich um eine Vererbung, bei der Eigenschaften des ursprünglichen Elementes überschrieben werden. Ein *Merge* führt dazu, dass gleichnamige Klassen in unterschiedlichen Paketen existieren, da Pakete erweitert werden. Ist letztere Eigenschaft nicht erwünscht, so ist das *Combine* zu verwenden. Auch bei dieser Beziehungsart können Elemente eines Paketes in einem neuen Paket mit weiteren Elementen kombiniert werden. Bei einem *Combine* werden jedoch neue Pakete definiert, die eine vollständige Kopie aller Elemente des ursprünglichen Paketes enthalten. Die Beziehung zwischen dem ursprünglichen und dem neuen Paket geht im Gegensatz zum *Merge* verloren. Das neue Paket steht für sich allein und beinhaltet sowohl die neue als auch alle Eigenschaften des ursprünglichen Paketes. Die drei genannten Beziehungstypen können mittels Stereotypen an den Abhängigkeitsbeziehungen von Paketen annotiert werden. Im Informationsmodell wird nur auf die „Import“ Beziehung zurückgegriffen. Die Notation für eine importierte Klasse erfolgt über Angabe des ursprünglichen Paketes unterhalb des Klassennamens (from *Paketname*).
5. *Klasse* (engl. Class): Besitzt ein Typ Objekte als Instanzen, so bezeichnet man ihn als Klasse. Sie kann Attribute (engl. Property) und Operationen (engl. Operation) besitzen und an einer Vererbungsbeziehung beteiligt sein. Ist eine Klasse abstrakt, kann sie keine direkten Instanzen besitzen. Operationen werden in dieser Arbeit nicht modelliert, da der Fokus auf den Objekten, deren Attribute und Assoziationen liegt.
6. *Assoziation* (engl. Association): Eine semantische Verbindung zwischen zwei Instanzen ist ein Assoziation. Sie besitzt genau zwei Enden, die durch Eigenschaften (engl. Properties) einschließlich eines Typs repräsentiert werden. Assoziationen können sowohl durch Angabe einer Navigierbarkeit als auch mittels Kardinalitäten an ihren Enden näher spezifiziert werden. Die Navigierbarkeit wird in den folgenden Modellen nicht verwendet, da nur die Strukturen und Eigenschaften von Objekten relevant sind.
   Führen die Klassen keine gleichwertige Beziehung, sondern eine „Ganzes-Teile“-Hierarchie, handelt es sich um eine *Aggregation*. Eine Aggregation, bei der die Teile vom Ganzen existenzabhängig sind und ein Teil genau einmal in einem Ganzen enthalten sein muss, bezeichnet man als *Komposition*.

Die Modellierung erfolgt mit Hilfe des Modellierungswerkzeuges Rational Rose (IBM), unter Verwendung von UML Klassendiagrammen. Abbildung 11 veranschaulicht die Notation.

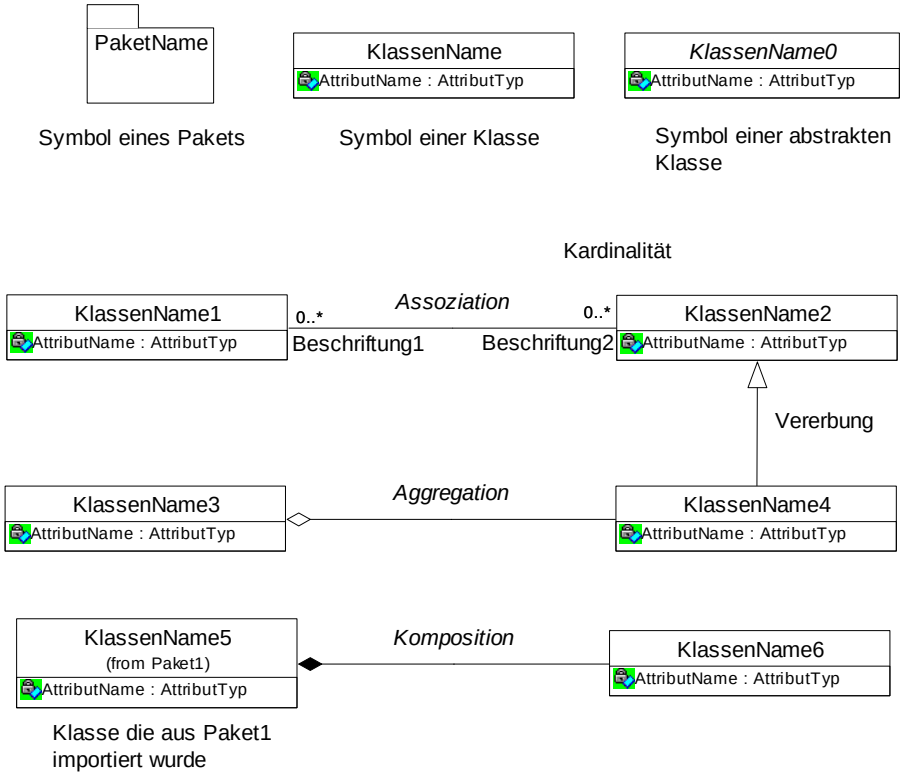


Abbildung 11: Notation für Modelle dieser Arbeit

3.2.3 Design Prinzipien

Das Kapitel der theoretischen Grundlagen soll durch die Einführung einiger Design Prinzipien abgerundet werden, die die Qualität eines Modells steigern und bei dessen Erstellung unterstützen.

Prinzipien sind Grundsätze, die dem Handeln als theoretische Grundlage zugrunde gelegt werden. Sie sind allgemeingültig, abstrakt und allgemeiner Art. Prinzipien werden aus der Erfahrung und Erkenntnis hergeleitet und durch sie bestätigt oder widerlegt [Ba98].

3.2.3.1 Prinzip der Abstraktion

Mit das wichtigste Prinzip, dass der Modellierung der Teilmodelle zugrunde liegt, ist das Prinzip der Abstraktion. Sie ist die Verallgemeinerung, das Absehen vom Besonderen und Einzelnen, das Loslösen vom Dinglichen. Dementsprechend definiert sich das Abstrahieren, als das Abgehen vom Konkreten, das Herausheben des Wesentlichen aus dem Zufälligen bzw. das Erkennen gleicher Merkmale [Ba98].

Mit Hilfe von Abstraktionsebenen, lassen sich Abstufungen der Abstraktion bezeichnen, wobei die Schwierigkeit sowohl allgemein, als auch bei den in dieser Arbeit entwickelten Modellen, in der Wahl der geeigneten Abstraktionsebene liegt. Es ist äußerst anspruchsvoll, aus vielen konkreten Tatsachen, das Wesentliche zu isolieren.

Die Vorteile der Anwendung des Abstraktionsprinzips, liegen in dem Erkennen, Ordnen, Klassifizieren und Gewichten von wesentlichen Merkmalen. Das Erkennen allgemeiner Charakteristika und das Trennen des Wesentlichen vom Unwesentlichen erleichtern die Betrachtung komplexer Sachverhalte.

Einen ersten Rahmen für die Abstraktion in dieser Arbeit liefert die M1 Ebene der Metamodell Hierarchie von MOF und UML. Die Wahl des Abstraktionsgrades der einzelnen M1 Modelle, wird dabei an den Anforderungen der Modell-Stakeholder ausgerichtet.

3.2.3.2 Grundsätze ordnungsmäßiger Modellierung

Die Grundsätze ordnungsmäßiger Modellierung fokussieren insbesondere auf die Qualität eines Informationsmodells und liefern Prinzipien, die bei der Entwicklung eines solchen unterstützend beitragen.

Becker et al. [BRS95] stellen fest, dass bei der Informationsmodellierung nicht mehr die Erstellung eines Informationsmodells die alleinige Zielsetzung ist, sondern auch dessen Anschaulichkeit, Verständlichkeit und Qualität an Bedeutung gewinnen. Semantische Gestaltungsempfehlungen zur bedarfsgerechten Informationsmodellierung hätten dabei im Gegensatz zur Darstellung syntaktischer Fragestellungen eine vergleichsweise geringe Problematisierung in der bisherigen Literatur erfahren. Mit Hilfe so genannter Grundsätze ordnungsmäßiger Modellierung (GoM) sollen bindende, aber auch flexible Gestaltungsempfehlungen für die Informationsmodellierung aufgestellt werden, so dass ein Rahmen vorgegeben wird, der die Modellerstellung unterstützt. Der Nutzen von GoM wird dabei laut Becker et al. [BRS95] in der Bewertung der Modellqualität, in der Verbesserung der Modellvergleichbarkeit und der Modellintegration gesehen.

Die GoM [BRS95], die in Anlehnung an die Grundsätze ordnungsmäßiger Buchführung entwickelt wurden, werden durch insgesamt sechs allgemeine Grundsätze vorgegeben. Diese lauten:

1. *Grundsatz der Richtigkeit*: Es wird unterschieden zwischen der syntaktischen und der semantischen Ausprägung. Ein Modell ist *syntaktisch* richtig, wenn es vollständig und konsistent gegenüber dem ihm zugrunde liegenden Metamodell ist, d.h. die verwendeten Informationsobjekte und Notationsregeln im Metamodell definiert sind. Die *semantische* Richtigkeit bemisst sich dahingegen an der Struktur- und Verhaltenstreue des Modells gegenüber dem abgebildeten Objektsystem. Sie orientiert sich an der Abbildungsfunktion. Die Forderung nach Widerspruchsfreiheit innerhalb des Modells, sowie zu anderen Modellen, zählt ebenfalls zur semantischen Richtigkeit.
2. *Grundsatz der Relevanz*: Unter Berücksichtigung der mit der Modellierung verbundenen Ziele sind die in einem Modell enthaltenen Elemente und Beziehungen genau dann relevant, wenn der Nutzeffekt der Modellverwendung sinken würde, falls das Modell weniger Informationen enthalten würde. Dabei ist zwischen der Auswahl des Objektsystemausschnitts und den für diesen Ausschnitt gewählten Objekte des Modellsystems zu unterscheiden (siehe Abbildung 12). Relevante Modellbestandteile müssen sich dabei nicht zwingend auf die geeignete Modellierung des Realweltausschnittes beziehen, sondern können relevant sein, weil sie die Verständlichkeit des Modellnutzers fördern [RS02].

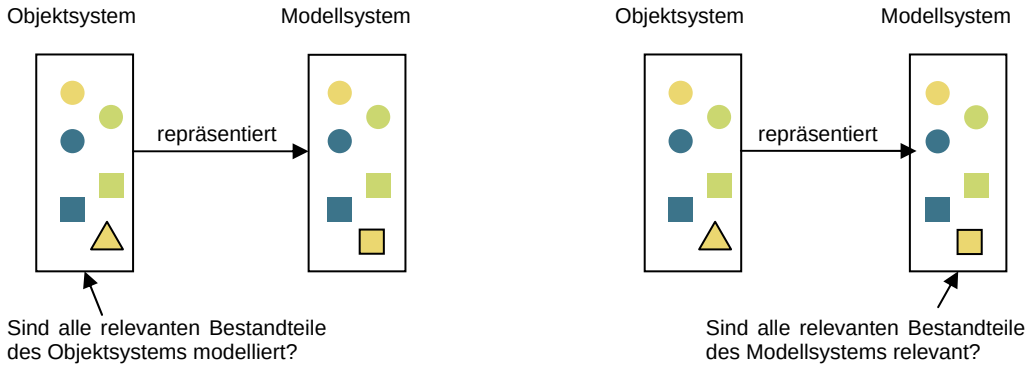


Abbildung 12: Objektrelevanz des Objekt- (links) und Modellsystems (rechts) aus [Be00]

3. *Grundsatz der Wirtschaftlichkeit*: Die Modellierung soll einer generellen betriebswirtschaftlichen Maxime (Erstellungsaufwand, Persistenz, Flexibilität) unterworfen werden, so dass der Modellierungsintensität eine obere Grenze gesetzt werden kann. Problematisch ist an dieser Stelle jedoch die fehlende Theorie einer spezifischen Modellierungskostenrechung und Leistungsrechnung.

4. *Grundsatz der Klarheit*: Zum Grundsatz der Klarheit zählen Aspekte der Strukturiertheit, Übersichtlichkeit oder Lesbarkeit. Dabei wird die Klarheit eines Modells et al. durch die graphische Anordnung der Informationsobjekte determiniert.
5. *Grundsatz der Vergleichbarkeit*: Bei der Vergleichbarkeit wird wie bei der Richtigkeit zwischen einer syntaktischen und einer semantischen Vergleichbarkeit unterschieden. Die Kompatibilität von mit unterschiedlichen Methoden erstellten Modellen beschreibt die *syntaktische* Vergleichbarkeit. Es wird gefordert, dass die den Methoden zugrunde liegenden Metamodelle integriert werden, um die Konsistenz der Modelle zu gewährleisten. Wird dahingegen die inhaltliche Vergleichbarkeit von Modellen diskutiert, so zählt dies zur *semantischen* Vergleichbarkeit. Ein Bespiel ist der Vergleich von Ist- und Sollmodellen.
6. *Grundsatz des systematischen Aufbaus*: Findet eine Modellierung in getrennten Sichten statt, so geht die Notwendigkeit der Integration der einzelnen Sichten einher. Der Grundsatz des systematischen Aufbaus fordert, dass erstens die Existenz einer auf einem Sichten übergreifenden Metamodell basierenden Architektur erforderlich ist, die einen strukturierenden Rahmen für die Beschreibungssichten bildet. Zweitens müssen Sichten orientierte Sachverhalte immer im Kontext mit den andern Sichten gesehen werden.

Die Grundsätze der Modellierung können oftmals widersprüchlich zueinander stehen. Beispielsweise können methodische Erweiterungen, die eine semantisch mächtigere Abbildung des Objektsystems erlauben (syntaktische Richtigkeit), dem Grundsatz der Klarheit widersprechen, wenn dadurch das Modell an Transparenz verlieren würde. Je nach Modellziel sind die Grundsätze und ihre Einhaltung zu gewichten.

Dem aufgeführten sechs Grundsätzen liegt der abbildungsorientierte Modellbegriff zugrunde (vgl. 3.1). Sollen die Grundsätze auf konstruktionsorientierte Modelle überführt werden, so bedarf es zwei weiterer Grundsätze, die im Folgenden benannt werden.

Zunächst ist der Grundsatz der Richtigkeit für konstruktionsorientierte Modelle nicht tragbar, da diese Modelle eine Interpretationsleistung des modellierenden Individuums darstellen und somit eine Überprüfung der Realitätsentsprechung nicht möglich ist. Da ebenso relevante Objekte oder Beziehungen der Kreativität des Modellierenden entspringen können und nicht zwingend in der gegenwärtigen Realität zu finden sind, ist der Grundsatz der Relevanz ebenfalls nicht tragbar. Folglich werden Richtigkeit und Relevanz durch den Grundsatz der *Konstruktionsadäquanz* ersetzt. Dieser Grundsatz verlangt zunächst den Konsens der Modellersteller, in Verbindung mit Modellnutzern, über das im Modell zu repräsentierende Problem. Der Zweck und Nutzen (Anforderung und Ziele) eines Modells ist Mittelpunkt der Bewertung [Be00].

Der zweite Grundsatz ist die *Sprachadäquanz*. Sie fokussiert die Beziehung zwischen Modellsystem und dem gewähltem Metamodell, wobei zwischen Spracheignung und Sprachrichtigkeit (entspricht der syntaktischen Richtigkeit) differenziert wird. Zur Eignung einer Sprache zählen dabei die semantische Mächtigkeit, Formalisierungsgrad und Sprachverständlichkeit.

Soll die Qualität eines Modells bestimmt werden, so sind die aufgeführten Grundsätze im Bezug auf die Umsetzung im Modell zu überprüfen. Im Anschluss an die Erläuterung des entwickelten Informationsmodells, wird in Abschnitt 5.3 dieses Vorgehen angewandt.

3.2.3.3 Muster

Muster (engl. pattern) geben eine bewährte generische Lösung für ein häufig wiederkehrendes, in bestimmten Situationen auftretendes Entwurfsproblem bei der Entwicklung von Modellen an. Sie unterstützen die Wiederverwendung von Lösungen, dokumentieren existierende und erprobte Entwurfserfahrungen und benennen bzw. erklären wichtige Entwürfe. Darüber hinaus helfen Muster bei der Auswahl von Entwurfsalternativen, können schneller zu einem richtigen Entwurf verhelfen und bieten ein gemeinsames Entwurfsvokabular und Verständnis für eine Gruppe von Entwicklern [Ba00].

3.2.3.3.1 Softwarepattern

Softwarepatterns, auch als Entwurfsmuster bezeichnet, sind bereits bei der Analyse und Konstruktion von objektorientierten Modellen für Software-Architekturen eines Anwendungssystems verbreitet. Nach Gamma et al. [Ga01] lässt sich die Qualität eines objektorientierten Systems nach der Sorgfäl-

tigkeit bewerten, welche die Entwickler der Zusammenarbeit zwischen den Objekten gewidmet haben. Eine Architektur kann unter Einsatz von Mustern einfacher, weniger umfangreich und sehr viel leichter verständlich sein.

Entwurfsmuster von Software-Architekturen lassen sich jedoch nicht auf die Entwicklung des in dieser Arbeit zu erarbeitende Informationsmodellen übertragen, da sie insbesondere für die Wiederverwendung von Funktionalitäten eines Systems eingesetzt werden [La04]. Das Informationsmodell beschränkt sich jedoch auf die Identifikation relevanter Objekte sowie deren Attribute und gegenseitigen Beziehungen.

3.2.3.3.2 Architekturpattern

Nach Gartner [La04] schlagen Architekturpatterns die Brücke zwischen der Geschäfts- und der Technologiearchitektur und stellen architektonische Konstrukte dar. Sie beschreiben, welche Technologieservices benötigt werden um das Geschäft zu unterstützen und wie diese Services miteinander interagieren. Muster können auf mehreren Abstraktionsebenen definieren werden, so dass zwischen den *conceptual* (deutsch konzeptionell), *logical* (deutsch logisch) und *physical* (deutsch physisch) Patterns unterschieden wird.

Conceptual Patterns sind high-level Strategien, welche die globalen Eigenschaften von Technologien und groß angelegten Komponenten beschreiben, die das Unternehmen unterstützen. Ein Beispiel sind Muster für die Struktur der Anwendungssysteme. In einem „monolithic“ Ansatz erbringt ein einziges, stark integriertes System die meiste Funktionalität für das Unternehmen, wohingegen in einem „suite“ Ansatz eine Gruppe von integrierten Anwendungssystemen die Funktionalität erbringt. Der „best-of-breed“ Ansatz umfasst dahingegen Anwendungssysteme für einzelne Geschäftsprozesse, die je nach Bedarf miteinander integriert werden.

Die *logical Patterns* beschreiben eine Ansammlung von Technologieservices zur Unterstützung einer bestimmten Geschäftsanforderung. Es definiert welche Technologieservices wie miteinander interoperieren, z.B. ein serviceorientiertes Modell.

Muster auf der niedrigsten Abstraktionsebene stellen die *physical Patterns* dar. Hierbei handelt es sich um eine Sammlung von Technologiekomponenten, deren Implementierung spezifiziert wird. Beispielsweise werden alle physischen Bausteine (Datenbank, Webserver, Client, Firewall etc.) beschrieben, die für das Szenario „Remote Transaction Processing“ benötigt werden.

Die Architekturpatterns beschreiben somit Strukturen der Architektur eines Unternehmens, sie helfen jedoch nicht bei der Identifikation relevanter Objekte bzw. deren Beziehungen.

3.2.3.3.3 Integrationspattern

Einen anderen, die Unternehmensmodellierung fokussierenden Ansatz beschreiben Kühn und Karagiannis mit den von ihnen definierten Integrationspatterns [Ec05]. Ihre zugrunde liegende Überzeugung ist, dass es nicht *die* Methode in der Unternehmensmodellierung gibt und auch nicht geben wird. Ihrer Meinung nach wird eine Kombination und Integration von unternehmensspezifischen Bündeln situations- und problemspezifischer Methoden und Methodenfragmenten benötigt. Mit Hilfe der Integrationspattern soll die Definition und Integration dieser Methoden ermöglicht werden.

Da das in dieser Arbeit entwickelte Informationsmodell jedoch einen integrativen Ansatz für ein Unternehmen verfolgt, finden die Integrationspatterns keine Verwendung.

Nachdem das zweite und dritte Kapitel in die thematischen und theoretischen Grundlagen der Arbeit eingeführt haben, kann mit der eigentlichen Aufgabenstellung im nächsten Kapitel begonnen werden.

Kapitel 4

Analyse bestehender Diagramme und Karten

Im Rahmen des Forschungsprojekts Softwarekartographie wurden in Zusammenarbeit mit mehreren großen Unternehmen verschiedene Softwarekarten untersucht und erste Klassifikationen von Kartentypen und Sichten erstellt [LMW05b]. Entwickelte Konzepte und Begriffe wurden von Lankes et al. [LMW05b] in die Begriffswelt des IEEE 1471 eingeordnet und in einem konzeptuellen Modell zur Beschreibung von Anwendungslandschaften miteinander in Beziehung gesetzt. In einem ersten Schritt sollen in dieser Arbeit vier repräsentative Softwarekarten der HVB Systems nach entwickelten Kriterien analysiert werden.

Der folgende Abschnitt 4.1 geht zunächst auf die Softwarekarten als Architekturbeschreibung ein und liefert den Rahmen für das anschließende Softwarekarten Reengineering in Abschnitt 4.2. Der am IT-Bebaungsplan exemplarisch durchgeführten Analyse (siehe 4.2.2) folgt der konzeptuelle Entwurf eines Visualisierungsmodells für die Softwarekartographie (siehe 4.3). Ein Resümee, dass die in der Analyse gewonnenen Erfahrungen schildert, aufgetretene Problematiken darlegt, sowie Ergänzungen und Anregungen für das derzeit im Forschungsprojekt entwickelte Softwarekartographie-Werkzeug erfasst, bildet in Abschnitt 4.4 den Abschluss des vierten Kapitels.

4.1 Softwarekarten als Architekturbeschreibung

In der thematischen Einführung dieser Arbeit, wurde erläutert, dass der Hauptbeitrag der Softwarekartographie zum Architekturmanagement in der Bereitstellung einer Methodik zur Dokumentation der Architektur von Anwendungslandschaften bzw. der Unternehmensarchitektur besteht (vgl. Abschnitt 2.3). Dieser Abschnitt soll entwickelte Begriffsdefinitionen[14] und *best-practices* einführen. Sie liefern die Basis für die in Abschnitt 4.2 durchgeführte Analyse der bestehenden Softwarekarten der HVB Systems und schaffen gleichzeitig die ersten Ansätze für das in Abschnitt 4.3 entwickelte Visualisierungsmodell.

Das in Abbildung 13 abgebildete Modell stellt die Zusammenhänge der im Anschluss eingeführten Begriffe und der durch sie bezeichneten Konzepte dar. Es handelt sich dabei um das im Rahmen des Forschungsprojektes Softwarekartographie angepasste konzeptuelle Modell des IEEE Standards 1471-2000. Für nähere Erläuterungen bezüglich der Adaption des IEEE 1471 für die Softwarekartographie sei auf [LMW05b] verwiesen.

Am Begin einer Dokumentation bzw. Architekturbeschreibung (engl. *Architectural Description*) stehen stets Personen, die einen Informationsbedarf bezüglich eines Systems besitzen, der mit der zu erstellenden Dokumentation gedeckt werden soll. Nach dem IEEE 1471 Standard werden diese Individuen als *Stakeholder*[15] bezeichnet, z.B. eine Person, ein Team oder eine Organisation. Im Umfeld der Softwarekartographie fallen unter die Rolle des Stakeholders alle diejenigen, die sowohl positiv als auch negativ durch die Softwarekarte beeinflusst werden (vgl. [HVB04c]). Positiv in dem Sinne, dass ihre *Questions* (deutsch Fragestellungen) und *Concerns* (deutsch Interessen) durch die Softwarekarten beantwortet werden (vgl. Abbildung 13). Negativ kann ein Stakeholder betroffen sein, falls beispielsweise die Softwarekarte Informationen beinhaltet, die ihn in seinem Handeln einschränken. Ein Beispiel für eine negative Beeinflussung ist eine Softwarekarte, die Programmierern von neuen Anwendungssystemen vorschreibt, dass nur die angegebenen standardisierten Produkte bei der Softwareentwicklung eingesetzt werden können.

[14] Im weitern Verlauf der Arbeit werden die englischen Fachbegriffe verwendet, um den Kontext des IEEE Standards zu verdeutlichen.

[15] Ein Stakeholder bekundet ein Interesse an etwas, er „hält die Stange“. Dies kann sowohl positiv als auch negativ gemeint sein: mit den Stangen (norddeutsch „staken“), einem Stück Holz, wurde im wilden Westen ein Stück Land abgesteckt, an dem derjenige somit sein Interesse bekundete [HVB04c].

Questions und *Concerns* des Stakeholders werden vom Analytiker (engl. *Analyst*) entgegengenommen und falls nötig, verfeinert oder neu formuliert, um bestimmte *Concerns* besser zu erfassen. Anschließend baut der Analyst verschiedene *Views* (deutsch Sichten) auf bzw. wählt sie aus (siehe Abbildung 13).

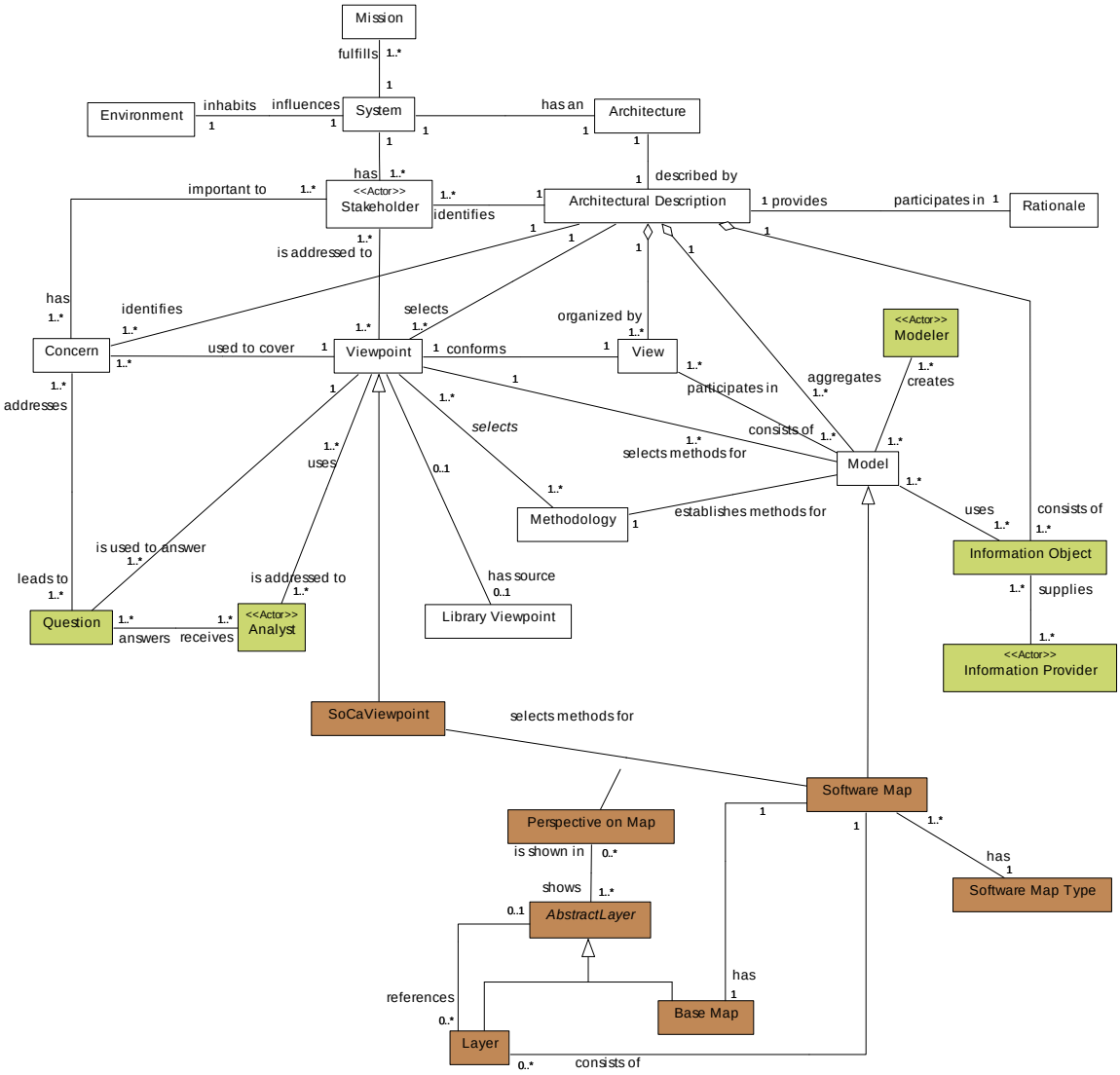


Abbildung 13: Ausschnitt des erweiterten konzeptuellen Modells des IEEE 1471 nach [LMW05c]

Views stellen dabei eine Repräsentation des gesamten Systems aus der Perspektive von assoziierten *Concerns* dar. Für die Konstruktion und die Nutzung einer bestimmten *View* werden wiederum so genannte *Viewpoints* (deutsch Standpunkt, Gesichtspunkt) in Form eines Musters oder einer Vorlage spezifiziert. Auf diese Weise findet eine Trennung der Definition (*Viewpoint*) und der konkreten Darstellung (*View*) statt. Da eine Architekturbeschreibung, den *Concerns* mehrerer Stakeholder gerecht werden soll, folgt die Notwendigkeit der Verwendung mehrerer *Views* für eine ausreichende Beschreibung des Systems. Für Softwarekarten lassen sich Dinge, die im normalen *Viewpoint* allgemein gehalten werden, mit Hilfe der Spezialisierung des *SoCaViewpoints* (Softwarekarten *Viewpoint*) genauer spezifizieren, um insbesondere das Schichtenprinzip von Softwarekarten adäquat abzubilden. Damit *Views* nicht für jede Architekturbeschreibung neu zu erfinden sind, bietet sich eine Wiederverwendung von *Viewpoints* mittels einer *Library Viewpoint* an (vgl. Abbildung 13).

Mit Hilfe der *Views* und *Viewpoints* kann der Analyst die durch sie definierten Modelle (z.B. Softwarekarten) vom Modellierer (engl. *Modeler*) erstellen lassen und sie zur Beantwortung der Stakeholder *Questions* nutzen. Der *Modeler* ist zusätzlich zur Modell- /Kartenerstellung auch für deren dauerhafte Pflege zuständig.

Der *Modeler* stützt sich bei seiner Tätigkeit auf die vom Informationslieferanten (engl. *Information Supplier*) bereitgestellten, konsolidierten und abgestimmten Informationsobjekte (engl. *Information Item*). Für die Erstellung einer Softwarekarte, die beispielsweise den Zusammenhang von Geschäftsprozessen und den sie unterstützenden Anwendungssystemen darstellt, sind die für das Modell benötigten Informationsobjekte, wie z.B. Geschäftsprozesse und Anwendungssysteme. Die Beschaffung der Informationsobjekte stellt dabei einen bedeutenden Kostenfaktor bei der Erstellung der Dokumentation einer Anwendungslandschaft dar.

Die Aufgaben des Analysten und des *Modelers* ist mit denen eines Kartographen zu vergleichen. Nach Hage et al. [HGM02] erzeugt ein Kartograph gewissermaßen einen Kommunikationskanal zwischen den Fachleuten, beispielsweise den Anwendungssystemspezialisten, und den Anwendern von Informationen, z.B. den Vorständen. Die Vorraussetzung für den Erfolg dieser Aufgabe ist dabei einerseits ein ausreichender Einblick in die Informationen des Fachmanns einschließlich einer geeigneten Informationsaufbereitung und andererseits das Verständnis für die Informationsbedürfnisse und die Auswertfähigkeit des Benutzers.

Da ein Modell zu mehreren *Views* gehören kann und zu jedem *View* genau ein *Viewpoint*, kann jedes Modell in mehreren *Viewpoints* verwendet werden. Um entsprechende Methodiken zu spezifizieren, bedarf es nun einem abstrakten Gegenstücks zum Modell, analog des Verhältnisses *Viewpoint* zu *View*. Es handelt sich um die so genannte Methodik (engl. *Methodology*), die Methoden und Techniken zur Konstruktion und Nutzung des entsprechenden Modells definiert (vgl. Abbildung 13).

Da Softwarekarten die Möglichkeit bieten, je nach Informationsbedarf Schichten (engl. *AbstractLayers*) ein- oder auszublenden, muss es eine Möglichkeit geben festzuhalten, welche Schichten eines *Viewpoints* für eine bestimmte Ansicht ein- bzw. ausgeblendet werden. Diese Aufgabe übernimmt die Kartenperspektive (engl. *Perspective on Map)*, wobei zu einem SoCaViewpoint mehrere Perspektiven eingesetzt werden können.

Bei einer Softwarekartenschicht (*AbstractLayer*) handelt es sich entweder um den Kartengrund (engl. *Base Map*) oder eine darauf abgebildete Schicht (engl. *Layer*). Der Kartengrund ist die unterste Schicht einer Softwarekarte und gibt den Rahmen vor, anhand dessen weitere Schichten aufgetragen und Kartenelemente positioniert werden können (siehe Abbildung 13). Laut Lankes et al. [LMW05a] besitzen Softwarekarten im Gegensatz zu geographischen Karten keinen eindeutigen Kartengrund (z.B. Längen- und Breitengrade), sondern verorten ihre Elemente nach unterschiedliche Kriterien und Merkmalen.

Im folgenden Abschnitt werden die soeben eingeführten Konzepte für die Analyse bestehender Diagramme und Softwarekarten der HVB Systems angewendet, so dass die bislang noch recht abstrakten Begrifflichkeiten anhand von Praxisbeispielen untermauert werden.

4.2 Softwarekarten-Reengineering

Dieser Abschnitt beschreibt das Vorgehen für ein *Softwarekarten-Reengineering*[16], welches die Entwicklung von *Viewpoint*-Beschreibungen basierend auf bestehenden Softwarekarten beschreibt. In der vorliegenden Arbeit wurden vier repräsentative Softwarekarten der HVB Systems ausgewählt, die nach einem erarbeiteten Schema analysiert wurden. Der nächste Abschnitt beschreibt zunächst, welcher Konzepte der Softwarekartographie Architekturbeschreibung für die Kartenanalyse verwendet werden. In Abschnitt 4.2.2 wird anschließend die Analyse exemplarisch für den IT-Bebauungsplan vorgestellt. Die Analysen der Building Block Landkarte, des Softwarearchitektur-Bebauungsplanes und der Technologiesets sind in Anhang A tabellarisch aufgeführt.

[16] Vgl. System-Reengineering (Verbesserung der Struktur und Verständlichkeit eines Systems) und Daten-Reengineering (Prozess der Analyse und Neuorganisation der Datenstrukturen um ein System verständlicher zu machen) [So01].

4.2.1 Vorgehensweise der Analyse

Die Kartenanalyse umfasst die im vorangehenden Abschnitt eingeführten Konzepte und beginnt dabei mit der Untersuchung des semantischen Karteninhaltes. Einleitend wird die Softwarekarte kurz beschrieben, ihr Zweck und Einsatzgebiet erläutert sowie Gründe für die Kartenerstellung erörtert. In einem nächsten Schritt werden durch die Angabe von beteiligten Personen und dessen Zuordnung zu definierten Rollen, einzelne *Concerns* und *Questions* hinterfragt, die für die Kartenerstellung und die äußere Form der Karte von besonderer Bedeutung sind. Für jede Softwarekarte wird dabei festgehalten, wie die Rollen besetzt sind, um wie viele Personen es sich je Rolle handelt (<10, <100, <1000, >1000) und welche *Concerns* und *Questions* zu den existierenden Softwarekarten geführt haben.

Nachdem im ersten Teil der Analyse das Augenmerk auf den durch die Karte repräsentierten Inhalt liegt, wird im zweiten Teil die äußere Form, d.h. Kartenstruktur und Gestaltungsmittel der Softwarekarte, untersucht. Der erste Analyseaspekt ist der Kartengrund, der die Verortungskriterien vorgibt und somit die Klassifizierung des Kartentyps ermöglicht. Kartentypen, die bislang im Softwarekartographie Projekt identifiziert wurden, sind neben den Karten ohne Kartengrund die Cluster-, Matrix-, Intervall- und Prozessunterstützungskarte (Definition siehe 2.3).

Da die Gestaltungsmittel einer Softwarekarte entscheidend zum Informationstransfer beitragen, ist es in einem nächsten Schritt notwendig, die in der Karte verwendeten Gestaltungsmittel (z.B. Signaturen, Punkte, Linie, Flächen, Text) sowie deren Gestaltungsvariablen (z.B. Höhe, Breite, Farbcode, Beschriftung) zu analysieren.

Häufig sind die Vielfalt und der Umfang der auf einer Softwarekarte zu visualisierenden Informationen sehr komplex, so dass geeignete Methoden für eine selektive Informationspräsentation eingesetzt werden müssen. Eine Lösung ist der Schichtenaufbau von Softwarekarten, bei dem auf einem Kartengrund mehrere Schichten nach Belieben des Betrachters ein- oder ausgeblendet werden können. Ob diese Technik bereits in den Karten der HVB Systems verwendet wird, oder ob andere Gestaltungsprinzipien z.B. mehrere *Viewpoints* eingesetzt werden, ist weiterer Bestandteil der Analyse.

Neben Karteninhalt und Kartenform ist die Informationsbeschaffung des *Information Suppliers* der letzte zu untersuchende Aspekt, wobei zum einen von Interesse ist, wie Informationen bei der initialen Kartenerstellung gewonnen werden und zum anderen, wie eine fortlaufende Kartenpflege von statten geht. Abschluss der Analysen bildet eine beispielhafte Ausprägung der jeweiligen Karte.

4.2.2 IT-Bebauungsplan Analyse

Der IT-Bebauungsplan dient der Analyse von Ist-, Soll- und Plan- Anwendungslandschaft. Er gewährleistet innerhalb von Projektarbeiten ein einheitliches Verständnis über die relevanten Anwendungssysteme eines Projektes und erlaubt Aussagen und Anregungen über die zukünftige Entwicklung der Anwendungslandschaft. Insbesondere findet der IT-Bebauungsplan Verwendung bei der Betrachtung von IT-Szenarien, welche eine mögliche Zukunftssituation innerhalb der IT darstellen. Die Anwendungslandschaft soll unter Einfluss unterschiedlicher Veränderungen auf relevante Aspekte untersucht werden. Aspekte können dabei strategisch (z.B. Outsourcing), wirtschaftlich (Plattform muss wartbar sein), fachlich (Prozesse, Organisationseinheiten) oder auch planerisch sein, wobei der IT-Bebauungsplan die fachlichen Aspekte visualisiert, in dem die IT-Szenarien mit den zugehörigen Anwendungssystemen einschließlich der Zuordnung zu Prozessen visualisiert werden.

In der Regel beziehen sich IT-Szenarien nur auf einen Teilbereich der gesamten Anwendungslandschaft, wobei sich IT-Szenarien häufig überschneiden, sie also auf gleiche Anwendungssysteme "einen Schatten" werfen. In diesen Fällen ist es besonders wichtig, aufgetretene Überschneidungen zu analysieren, um die gegenseitigen Einflüsse bzw. Abhängigkeiten zu identifizieren und mögliche Synergieeffekte in die Planung mit einzubeziehen. Synergien sind z.B. Workflow, Content-Management, Output-Management, Schnittstellentechnik und Einsatz von SAP Komponenten.

Ein Beispiel für ein IT-Szenario ist die Betrachtung der Anwendungslandschaft hinsichtlich des Einsatzes einer Standardsoftware, wie z.B. *Geos* für die Wertpapierabwicklung. Innerhalb dieses IT-Szenarios wird anhand der obigen Aspekte untersucht, welche Auswirkungen der Einsatz von Geos auf die Anwendungslandschaft besitzt. Beispielsweise kommen planerische Aspekte sehr stark zum Tragen:

- Welche Anwendungssysteme der Ist- Anwendungslandschaft würden durch Geos ersetzt werden und somit wegfallen?
- Welche weiteren Anwendungssysteme wären in Bezug auf Kommunikation über Geos Schnittstellen betroffen?
- Wie sähe die Soll- Anwendungslandschaft nach der Geos Integration aus?
- Auch wirtschaftliche Aspekte werden durch das IT-Szenario genauer betrachtet: Welche Anschaffungskosten entstehen bei Geos?
- Bringt eine Migration zu Geos hohe Ablösungskosten mit sich, so dass sich die Umstellung nicht rentieren würde?

Der IT-Bebauungsplan wird derzeit projektspezifisch erstellt und für die interne Kommunikation genutzt. Im Gegensatz zu der Building Block Landkarte (siehe A.1) liegt das Ziel nicht in der ständigen Aktualität und der vollständigen Prozessdarstellung mit zugehörigen Anwendungssystemen, sondern in der Visualisierung wesentlicher Informationen für ein konkretes Untersuchungsfeld. Die Informationen über Ist-, Plan- und Soll-Anwendungslandschaft erlauben Aussagen über die Umsetzung von gesetzten Zielen, so dass der IT-Bebauungsplan Wirtschaftsprüfungscharakter erhält. Abbildung 14 veranschaulicht die Zusammenhänge der zuvor erläuterten Informationsobjekte des IT-Bebauungsplans.

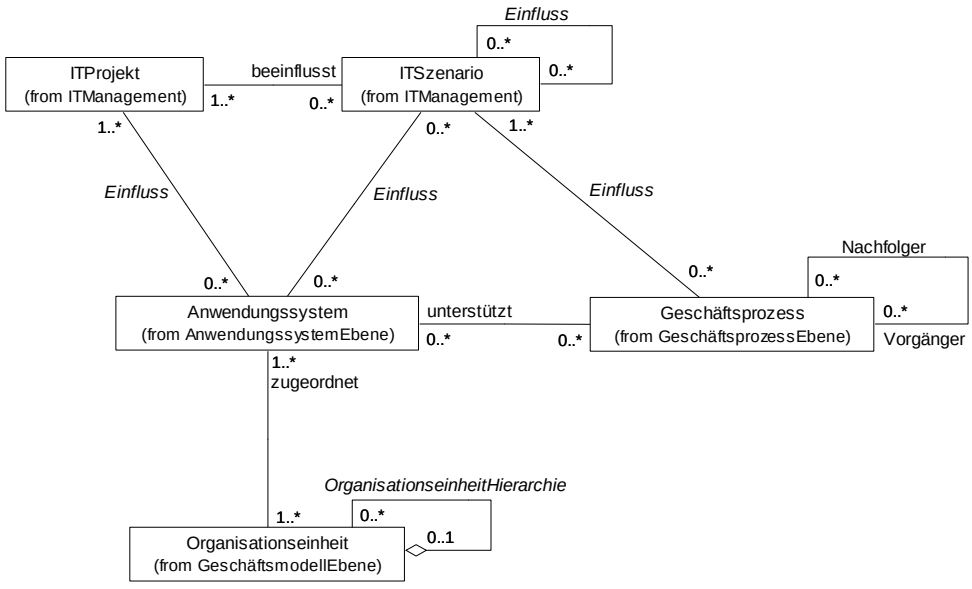


Abbildung 14: Informationsobjekte des IT-Bebauungsplans

4.2.2.1 Rollen

Stakeholder des IT-Bebauungsplans sind Architekten der Anwendungslandschaft, COOs, Wirtschaftsprüfer, Revisionsbeteiligte, Geschäftsführer der Kundenfirmen, Unternehmensberater und Geschäftsfelder der HVB Systems. Insgesamt nutzen bis zu hundert Personen die Softwarekarte. Die Rolle des Analysten und des *Modelers* nehmen bis zu zehn Architekten bzw. Berater ein, wohin gegen die Größenordnung bei dem *Information Supplier* bei bis zu tausend Personen liegt. Der Grund für diese große Anzahl liegt in den verstreuten Informationen. Nicht alle Informationen sind im Firmen Repository,

Architekturportal PASS[17] oder dem firmen ZAD (Zentraler Anweisungs-Dienst) vorliegen, sondern müssen aus verschiedenen Datenbeständen und mit Hilfe verschiedener Personen zusammen getragen werden. Häufig stellt sich dabei die Konsolidierung der Informationen als besonders schwierig und langwierig heraus, da widersprechende Daten aufeinander abgestimmt und konsolidiert werden müssen. An der Informationsbeschaffung sind Geschäftsfeld-, Building Block- und Prozessverantwortliche, Architekten und CIOs beteiligt. Es ist durchaus üblich, dass die genannten *Information Supplier* den Auftrag der Informationsbeschaffungsauftrag innerhalb der Unternehmenshierarchie nach unten weitergeben.

4.2.2.2 Concerns und Questions

Stakeholder *Concerns*, die mit Hilfe des IT-Bebauungsplanes verfolgt werden sollen, sind:

- Steuerbarkeit des Geschäftsfeldportfolios
- Ausnutzung von Synergieeffekten
- Schaffung einer effizienten Architektur
- Veranschaulichung der Geschäftsstrategieauswirkungen auf die Anwendungslandschaft

Aus diesen *Concerns* lassen sich *Questions* ableitet, die beispielsweise nach der IT-technischen Abhängigkeit der einzelnen Geschäftsfelder fragen. Zudem ist es von großem Interesse, ob operative Synergien zwischen Geschäftsfeldern ausgenutzt werden (z.B. einheitlicher Zahlungsverkehr oder Wertpapierabwicklung). Die Frage nach der end-to-end Optimierung von Geschäftsprozessen bzw. ihrer Durchgängigkeit ist ebenso wichtig wie die Frage nach der Geschäftsprozessunterstützung durch die Anwendungssysteme.

Die Aufgabe des Analysten besteht darin, aus den *Concerns* und *Questions* die benötigten Elemente der Softwarekarte festzulegen. In einigen Fällen ist es notwendig, weitere *Questions* aufzustellen, die mehr Klarheit über die *Questions* der Stakeholder schaffen. So kommt es beispielsweise zu der Frage, wieweit die Prinzipien der Kapselung und der Verwendung von Standards in der aktuellen Anwendungslandschaft umgesetzt werden. Auch der Einfluss, der aus der Geschäftsstrategie ableiteten IT-Szenarien auf die Anwendungslandschaft sind von besonderer Bedeutung. Aussagen über die Ist-, Soll- und Plan-Anwendungslandschaft sollen gemacht werden können um beispielsweise den Aufwand und Mehrwert eines Wechsels zu einer standardisierten Lösung abschätzen zu können.

Bei der gesamten Betrachtung ist der Status eines jeden Anwendungssystems zu beachten, um eventuelle Handlungsbedarfe oder Entscheidungsalternativen zu berücksichtigen.

4.2.2.3 Klassifikation nach Kartengrund

Anwendungssysteme werden anhand der Geschäftsprozesse auf der x-Achse verortet, so dass es sich beim IT-Bebauungsplan um eine Prozessunterstützungskarte handelt. Die y-Achse des IT-Bebauungsplans kann für die Zuordnung zu Produkten oder Märkten verwendet werden und wird durch eine vertikale Beschriftung vermerkt. Im vorliegenden IT-Bebauungsplan wurde auf diese Zuordnung verzichtet.

Zusätzlich zur Verortung anhand der Kriterien von x- und y-Achse spielt die relative Position der Anwendungssysteme eine Rolle, die aufgrund ihrer gemeinsamen Zugehörigkeit zu IT-Szenarien bzw. Projekte möglichst nah angeordnet werden.

4.2.2.4 Gestaltungsmittel

Abbildung 15 umfasst die Gestaltungsmittel des IT-Bebauungsplans, auf die im Folgenden eingegangen wird.

[17] PASS steht für „Prozesse, Architektur der IT, Strukturen, Standards“. Das Portal stellt Prozesse mit Organisationen und IT-Systemen nach gleichartigen Grundmustern dar.

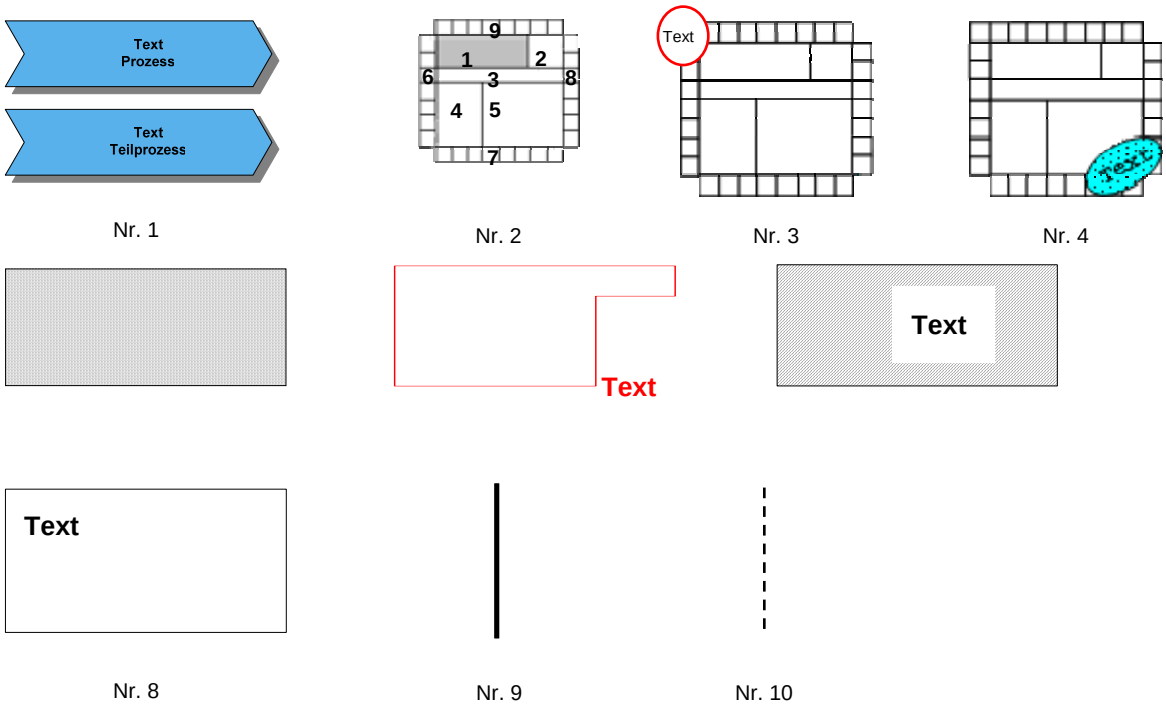


Abbildung 15: Gestaltungsmittel des IT-Bebauungsplans

Geschäftsprozesse und Teilprozesse (Nr. 1) werden mittels eines beschrifteten Chevrons dargestellt. Der Text enthält den Namen des Prozesses bzw. Teilprozesses und ist sowohl vertikal als auch horizontal zentriert ausgerichtet. Teilprozesse werden unterhalb der ihnen zugehörigen Prozesse angeordnet, welche entsprechend horizontal ausgedehnt werden.

Anwendungssysteme werden mittels einer symbolischen Signatur (Nr. 2) visualisiert. Die Beschriftung des Feldes (1) trägt den Namen des Anwendungssystems, Feld (2) enthält das entsprechende Kürzel. Handelt es sich um ein konkretes Teilsystem, so wird dessen Bezeichnung in Feld (3) eingetragen. Feld (4) wird mit der langen Bezeichnung des Anwendungssystems beschriftet. Die zugehörige Beschreibung kann in Feld (5) vorgenommen werden. Die Felder (6)-(9) können für die Angabe von Schnittstellen bzw. genutzte/angebotene Services verwendet werden, da diese Felder in den analysierten IT-Bebauunsplänen keine Verwendung fanden, wird an dieser Stelle keine genauere Erläuterung gegeben.

Die Symbolische Signatur des Anwendungssystems kann durch einen beschrifteten Kreis ergänzt werden (Nr. 3). Die Beschriftung wird innerhalb des Kreises sowohl vertikal als auch horizontal „zentriert“ ausgerichtet. Der Kreis repräsentiert dabei die Organisationseinheit (z.B. Geschäftsfeld d.h. Organisationseinheit grober Granularität), dem das Anwendungssystem aufgrund der stärksten Nutzung zugeordnet ist. Diese Zusatzinformation kann bei der Kartenpflege und -nutzung hilfreich sein.

Auf die gleiche Weise kann mittels einer beschrifteten Ellipse die technisch zugeordnete Abteilung (Organisationseinheit feiner Granularität) bzw. der Produktverantwortliche eines Anwendungssystems dokumentiert werden (Nr. 4). Die Beschriftung der Ellipse ist „zentriert“ entlang der Hauptachse und auch der Nebenachse platziert, der Text beginnt im linken und verläuft zum rechten Hauptscheitel.

Findet ein Anwendungssystem in mehreren Geschäftsprozessen Verwendung, so wird sein Nutzungsbereich mittels eines gepunkteten Rechtecks visualisiert (Nr. 5). Das Rechteck wird im Hintergrund des Anwendungssystem-Symbols platziert und sowohl horizontal als auch vertikal entsprechend des Verwendungsbereiches ausgedehnt. Eine Alternative für die Visualisierung des Anwendungssystems Verwendungsbereiches wäre die erneute Darstellung der Anwendung durch dessen Symbolik. Der Vorteil, der hier gewählten, auf Symbol-Redundanzen verzichtende Illustration liegt darin, dass dem Kartenbetrachter die Anzahl der verschiedenen Anwendungssysteme ersichtlich ist. Würde man dahingegen die Symbolik für das Anwendungssystem erneut auf die Karte auftragen, so

erschien der IT-Bebauungsplan aufgrund der hohen Symbolanzahl vielfältiger und komplexer. Ein Nachteil, der aus der Darstellung mittels der umgebenden Box resultiert, ist, dass der IT-Bebauungsplan fälschlicherweise komplett und aufgeräumt erscheint. Dass sich allerdings noch viele weitere Anwendungssysteme hinter den Prozessen verbergen, die aus Effizienz und Übersichts- Gründen nicht dargestellt wurden, ist dem Kartenbetrachter nicht auf Anhieb bewusst.

Das Gestaltungsmittel für IT-Szenarien ist ein Vieleck, das im Hintergrund der Anwendungssystem-Symbole platziert wird (Nr. 6). Um verschiedene, sich überschneidende IT-Szenarien visuell voneinander zu trennen, kann die Hintergrund- und die Rahmenfarbe beliebig gewählt werden. Die Beschriftung wird Platz optimierend innerhalb oder außerhalb des Vieleckes platziert und gibt den Namen des IT-Szenarios an.

Der Einflussbereich von Projekten wird ebenfalls mittels Vielecke, die im Hintergrund der Anwendungssystem-Symbole platziert werden, gekennzeichnet (Nr. 7). Das Vieleck umgibt alle Symbole der Anwendungssysteme, die für das Projekt eine Rolle spielen. Das Vieleck wird sowohl horizontal als auch vertikal entsprechend ausgedehnt. Die Beschriftung wird Platz optimierend innerhalb des Vieleckes platziert und gibt Hinweise auf das Projekt.

Gruppierungen von Anwendungssystemen nach bestimmten Merkmalen werden als Rechtecke im Hintergrund der Anwendungssystem-Symbole kennzeichnet (Nr. 8). Das Rechteck wird sowohl horizontal als auch vertikal entsprechend ausgedehnt, die Beschriftung wird innerhalb des Rechteckes sowohl vertikal als auch horizontal „oben“ ausgerichtet und betitelt das Gruppierungsmerkmal.

Schwarze, vertikale Linie (Nr. 9) kennzeichnet die Vermutung einer fachlich sauberen Trennung von Prozessen hinsichtlich der Building Blocks und geben Hinweis auf Schnittstellen. Bei diesem Gestaltungsmittel handelt es sich um eine lineare Signatur.

Eine weitere lineare Signatur wird für die Veranschaulichung von Hilfslinien eingesetzt. Hierbei handelt es sich um vertikale, gestrichelte Linien (Nr. 10), die das Erkennen der Zugehörigkeit von Anwendungssystemen zu Teilprozessen erleichtern.

Zeitliche Aspekte können zu verschiedenen Instanzen der Anwendungslandschaft führen, ein entsprechender Vermerk im Softwarekartentitel kann Aufschluss über den Aufnahmezeitpunkt geben (z.B. Bild der Anwendungslandschaft zum Zeitpunkt T).

4.2.2.5 Farbcodes (Schraffuren, Hintergründe, Farben für Rahmen)

Für die Veranschaulichung des Anwendungssystem-Status, werden im IT-Bebauungsplan unterschiedliche Farben eingesetzt (siehe Abbildung 16). Rot bedeutet für ein Anwendungssystem „wird abgelöst“, blau „wird eventuell abgelöst“. Lila kennzeichnet, dass für ein Anwendungssystem eine „teilweise Ablösung“ vorgenommen wird, grün bedeutet „bleibt, wird aber geändert/ Schnittstellen ändern sich“. Eine schwarze Symbolik signalisiert „bleibt, wird nicht abgelöst“.

Ergänzend zu dem Farbcode wird in der Kartenlegende das Datum der Statusbestimmung angegeben.

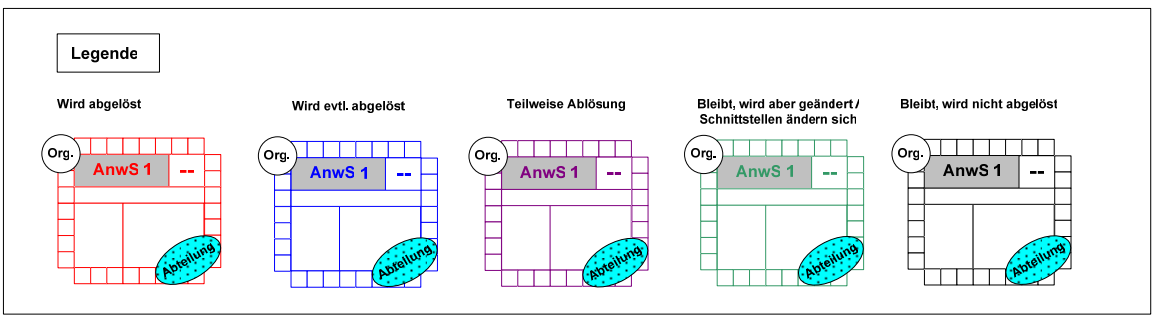


Abbildung 16: Legende des IT-Bebauungsplans

4.2.2.6 Schichtenprinzip

Der IT-Bebauungsplan ist ein statisches Dokument[18], dass gegebenenfalls in ein PDF oder JPG Format konvertiert werden kann. Folglich besteht er lediglich aus einer einzelnen Schicht, in der alle Informationen abgebildet werden. Es ist allerdings vorstellbar, dass bei einem zukünftigen IT-Bebauungsplan, der nicht händisch sondern automatisch generiert wird, mehrere Schichten definiert werden, um dem Benutzer die Möglichkeit zu bieten, einzelne Schichten ein- bzw. auszublenden. Die erste Schicht könnte hierbei den Kartengrund, basierend auf Prozesse und Teilprozesse, umfassen. Auf der nächsten Schicht könnten die Anwendungssysteme eingeblendet werden, die in einer dritten und vierten Schicht um die Organisations- und Abteilungsinformationen ergänzt werden könnten.

Abbildung 17 zeigt exemplarisch den IT-Bebauungsplan, wobei firmeninterne Informationen gelöscht wurden.

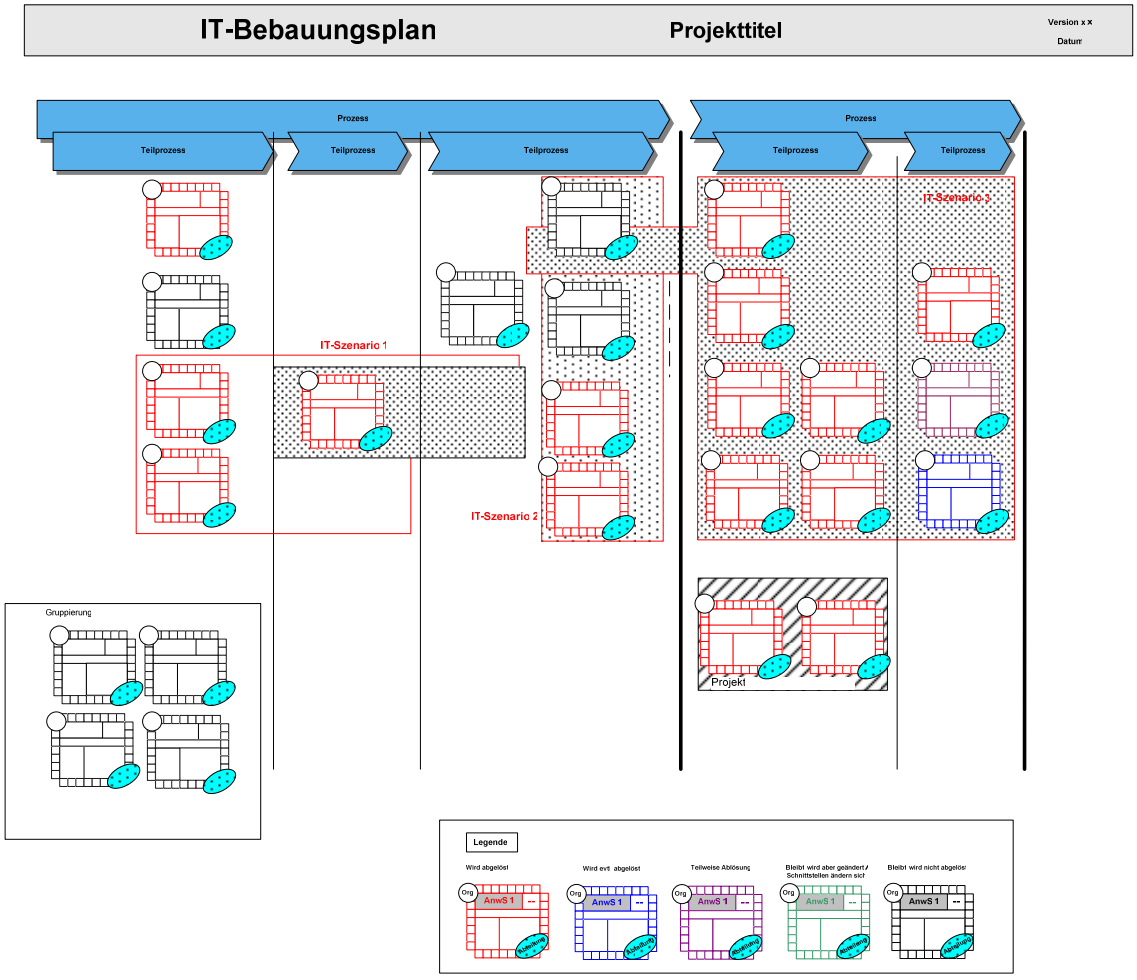


Abbildung 17: IT-Bebauungsplan

4.2.2.7 Viewpoint

Im Sinne des IEEE Standards 1471, ist der IT-Bebauungsplan ein Modell für die Architekturbeschreibung der Anwendungslandschaft. Der zugehörige *Viewpoint* lässt sich beispielsweise als „IT-Szenarien und Projekte der Anwendungslandschaft“ bezeichnen, da er das Zusammenspiel von Projekten und IT-Szenarien bezüglich deren Auswirkungen auf die Anwendungssysteme veranschaulicht. Dieser *Viewpoint* besteht aus einem einzigen Modell.

[18] Erstellt mit Microsoft Visio (Microsoft).

4.2.2.8 Initiale Kartenerstellung und anschließende Kartenpflege

Die Erstellung des IT-Bebauungsplans wird durch den Architekten vorgenommen. Er sammelt die für die Karte benötigten Informationen aus dem Unternehmens Repository, dem Architekturportal PASS oder ermittelt sie mit Hilfe der obig aufgelisteten *Information Supplier*. Anschließend werden alle Objekte händisch in Visio angelegt und platziert.

Je nach Projekt bestehen unterschiedliche Anforderungen hinsichtlich der Aktualität der IT-Bebauungspläne, so dass keine allgemeingültige Aussage über den Zeitraum der Kartenaktualisierung gemacht werden kann.

4.3 Konzeptuelles Visualisierungsmodell

Basierend auf dem Verständnis der Softwarekarten als eine Architekturbeschreibung und den Erkenntnissen der Analyse bestehender Softwarekarten der HVB Systems, soll in diesem Abschnitt ein konzeptueller Entwurf des Visualisierungsmodells der Softwarekartographie entwickelt werden.

In Abschnitt 3.1.4.2 wurde das Visualisierungsmodell auf der zweiten Ebene des symbolischen Modells eingeordnet. Aufgabe des Visualisierungsmodells besteht darin, Eigenschaften und Zusammenhänge der Visualisierungsobjekte zu definieren. Es kann zum einen für die Entwicklung eines implementierungsnahen Modells des Softwarekartographie-Werkzeugs verwendet werden, zum anderen fasst es Begriffe rund um die Softwarekarte zusammen. Folgend werden die Komponenten des in Abbildung 19 veranschaulichten Visualisierungsmodells beschrieben.

Das zentrale Objekt des Visualisierungsmodell ist die *Softwarekarte*, eine graphische Repräsentation der Anwendungslandschaft oder von Ausschnitten selbiger. Sollen Anwendungssysteme der Landschaft und deren Komponenten veranschaulicht werden, so kann es auch zu Darstellungen kommen, die eher technische Aspekte, wie beispielsweise Bausteine und Produkte von Anwendungssystemen, in Zusammenhang bringen. Eine Softwarekarte wird über die für einen Stakeholder relevanten und die Unternehmensarchitektur betreffenden Aspekte charakterisiert. Eigenschaften wie die Kartenbenennung, die das Kartenthema angibt, sowie Angaben der die Rollen einnehmenden Personen einschließlich ihrer *Concerns* und *Questions* geben Auskunft über die inhaltlichen Aspekte der Karte. Weitere, die Verwaltung der Karte betreffende und informierende Attribute, ergänzen die Beschreibung der Softwarekarte. Beispielsweise dokumentiert der Aufnahmezeitpunkt gegenüber dem Erstellzeitpunkt, auf welchen Datenbestand sich die Darstellung bezieht. Eine Karte, die im März eines Jahres erstellt wurde, kann sich durchaus auf einen Datenbestand im Januar beziehen, wenn die aktuelleren Daten noch nicht vollständig zur Verfügung stehen.

Die Notation einer Softwarekarte wird entsprechend dem IEEE Standard 1471 in einem *Softwarekartenviewpoint* definiert.

Der Aufbau einer Karte lässt sich laut Hake et al. [HGM02] durch formale und sachliche Merkmale beschreiben. Zu den formalen Merkmalen zählt zunächst die Kartenbenennung. Ein weiterer formaler Bestandteil der Karte ist das *Kartenfeld*, d.h. die Fläche, die den Karteninhalt enthält und vom so genannten *Kartenrahmen* begrenzt wird. Dieser ist eine streifenförmige schmale Fläche zwischen der inneren Kartenschnittlinie und einer äußeren Begrenzungslinie, an der der Kartenrand beginnt. Der Kartenrahmen kann optional für Angaben der Karte verwendet werden. Außerhalb des Kartenrahmens befindet sich der *Kartenrand*, welcher durch das meist rechteckige Kartenformat abgegrenzt wird. Größe und Form des Kartenrandes richten sich dabei nach dem Umfang und einer sinnvollen Verteilung der Kartenrandangaben wie beispielsweise die Kartenbenennung, Texte oder Legenden. In der *Legende* (Zeichenerklärung) werden die Darstellung der Kartenobjekte, sowie die Abkürzungen für Objektnamen erläutert, da bei der Vielzahl von Einzelheiten, die auf einer Karte zusehen sind, eine ausführliche Zeichenerklärung unerlässlich ist [Ko04].

Zu den sachlichen Kartenmerkmalen zählt als wichtigster Punkt der *Karteninhalt*. Dieser ist im syntaktischen Sinne die Summe der graphischen Darstellungen bzw. der dafür stehenden digitalen Daten und im semantischen Sinne die Gesamtheit der Objekte, für die die Graphik bzw. deren digitale Daten stehen. Um welche Informationen und Informationsobjekte es sich in der Karte handelt, wird in der Kartenbeschreibung dokumentiert. Die Visualisierung des Karteninhaltes erfolgt nach Lankes et al. [LMW05a] mittels mehreren Kartenebenen des Kartenfeldes. Die unterste Ebene bildet dabei in Anlehnung an die Kartographie der *Kartengrund*, der die Verortung der Kartenzeichen vorgibt. Auf dem

Kartengrund aufbauende *Kartenebenen* können verschiedene Aspekte der Anwendungslandschaft visualisieren (vgl. Abbildung 19). Kartenebenen können entweder den Kartengrund oder andere Ebenen referenzieren. Dieses Ebenenprinzip ermöglicht es, unterschiedliche *Questions* unter Wiederverwendung eines bestehenden Kartengrundes zu beantworten. Indem zusätzliche Ebenen genutzt werden, werden zusätzliche Informationen visualisieren. Die Klasse *AbstrakteKartenebene* bezeichnet alle Ebenen zusammen, die dann wiederum in Kartengrund und die restlichen Ebenen zerfallen.

Laut Lankes et al. [LMW05c] bieten Softwarekarten, als Mittel zur Komplexitätsreduktion von Darstellungen die Möglichkeit, bestimmte Schichten (vgl. AbstrakteKartenebene in Abbildung 19) ein- oder auszublenden. Eine derartige Kombination aus angezeigten und nicht angezeigten Schichten wird durch die *Kartenperspektive* definiert. Eine Legende wird stets einer Kartenperspektive zugeordnet, da in ihr alle auf den durch die Kartenperspektive definierten Ebenen angezeigten Kartenzeichen erläutert werden. Enthalten zwei Kartenperspektiven nur solche Ebenen, die sich lediglich durch Filterregeln unterscheiden, nicht jedoch durch die Art der Kartenzeichen, ist es möglich, dass ein und dieselbe Legende für mehrere Kartenperspektiven gültig ist. Beispielsweise hat eine Softwarekarte, die alle Anwendungssysteme eines Geschäftsprozesses visualisiert, die gleiche Legende wie die Softwarekarte, die nur die Anwendungssysteme des Geschäftsprozesses visualisiert, die abgelöst werden sollen.

Da die Gestaltung des Kartengrundes das äußere Erscheinungsbild einer Softwarekarte stark beeinflusst, lassen sich laut [LMW05a] Softwarekarten anhand der für die Verortung ihrer Inhalte verwendeten Prinzipien in *Softwarekartentypen* einteilen. In Abschnitt 2.3 wurden bereits die Kartentypen *Cluster-, Matrix-, Intervall-* und *Prozessunterstützungskarte* eingeführt (vgl. Abbildung 19).

Die Gestaltung einer Karte ist gekennzeichnet durch eine Kartengraphik, d.h. die Gesamtheit der für Karten aller Art typischen Darstellungsweisen, welche die Merkmale und Regeln aller graphischen Darstellungen umfasst. Jedes *Kartenzeichen* codiert Informationen und liefert für sich allein und aus der Beziehung zwischen den Zeichen mannigfaltige Aussagen über relevante Aspekte. Aspekte sind im Falle der Softwarekartographie *Informationsmodellelemente* (siehe Abbildung 19). In der UML Infrastructure [OMG03a] ist dabei ein Element „an element that is an abstraction drawn from the system being modeled“. Im Bezug auf das in Kapitel 5 entwickelte Informationsmodell handelt es sich dabei um die einzelnen Informationsobjekte.

Nach Hake et al. [HGM02] ergibt sich ein dreistufiger Aufbau dieses Zeichensystems. Die erste Stufe bilden die *graphischen Elemente*, die nach ihrer geometrischen Ausbreitung zu unterscheidenden *Punkte, Linien* und *Flächen* als Bausteine jeder Graphik. In die zweite Stufe fallen die *zusammengesetzten Zeichen*, d.h. spezifische Zusammenfügungen der graphischen Elemente zu höheren Gebilden. Zusammengesetzte Zeichen sind die *Signatur*, das *Diagram*, der *Halbton* und die *Schrift*. Graphische Elemente und zusammengesetzte Zeichen werden unter dem Oberbegriff kartographisches *Gestaltungsmittel* zusammengefasst (vgl. Abbildung 19). Auf der dritten Stufe stehen die graphischen Gefüge, die sich ergeben, wenn die Elemente und Zeichen bei jeweils bestimmten Objektarten typische graphische Strukturen erzeugen und damit in starkem Maße den Gesamteindruck der Karte bestimmen. Beispiele sind hierfür die Visualisierungen von Verkehrswege, Grenzen oder Gewässer.

Gestaltungsmittel besitzen ein oder mehrere *Gestaltungsvariablen*, durch deren Einsatz das Erscheinungsbild des Gestaltungsmittels auf bestimmte Weise variiert und somit Eigenschaften des zu repräsentierenden Informationsobjektes ausgedrückt werden können (siehe Abbildung 19). Gestaltungsvariablen können nach Hake et al. [HGM02] drei unterschiedliche Wirkungen hervorrufen. Zum einen dient es der objektiven Gliederung durch die differenzierte Darstellung nach Qualitäten und/oder Quantitäten der Objekte. Zum anderen haben Variablen Auswirkung auf die subjektive Bewertung durch Betonen oder Zurückdrängen. Zu guter Letzt, verstärkten sie die Anschaulichkeit auf der Basis von Assoziationen. Abbildung 18 zeigt beispielhafte Variationen des Gestaltungsmittels Rechteck.

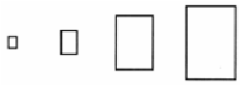
Bezeichnung der Variation	Ausgangszeichen	Beispiele der Variation
Größe		
Form		
Füllung		
Richtung		
Tonwert (unbunt, bunt)		
Farbe (bunt)		

Abbildung 18: Gestaltungsvariablen aus [HGM02]

Soll eine Softwarekarte aus bestehenden Daten automatisch generiert werden, so bedarf es einer Menge von *Regeln* die die kontextabhängige Gestaltung der kartographischen Darstellung gewährleistet [HGM02]. Regeln können dabei zu *Regelsätzen* zusammengefasst werden (vgl. Abbildung 19). Sie umfassen sowohl Regeln, die sich auf die Informations- und Beziehungsobjekt-Eigenschaften beziehen (z.B. ist Anwendungssystem eine Eigenentwicklung dann färbe es grün) als auch externen Angaben (z.B. soll Softwarekarte im Intranet veröffentlicht werden, so müssen Corporate Identity Vorschriften eingehalten werden). Darüber hinaus ist bei den Regeln zwischen solchen zu unterscheiden, die das semantische Modell verändern bzw. verletzten, und solchen, die dies nicht tun. Regeln können mathematisch definiert werden, beispielsweise Positionierungsregeln mittels Ungleichungen. Da es sich bei diesem Modell um das konzeptuelle und nicht um das implementierungsnahe Visualisierungsmodell handelt, wird an dieser Stelle auf die Arbeiten des sebis Lehrstuhls verwiesen [Seb04][19].

Die Klasse *Transformation* steht repräsentativ für den in Abschnitt 3.1.4.3 eingeführten Prozess. Sie stellt das Bindeglied zwischen den bei einer Transformation zu berücksichtigenden Regeln und den mittels entsprechenden Gestaltungsmitteln, auf einer Kartenebene zu repräsentierenden Informationsobjekten.

[19] Stichwort Visualisierungsmodell.

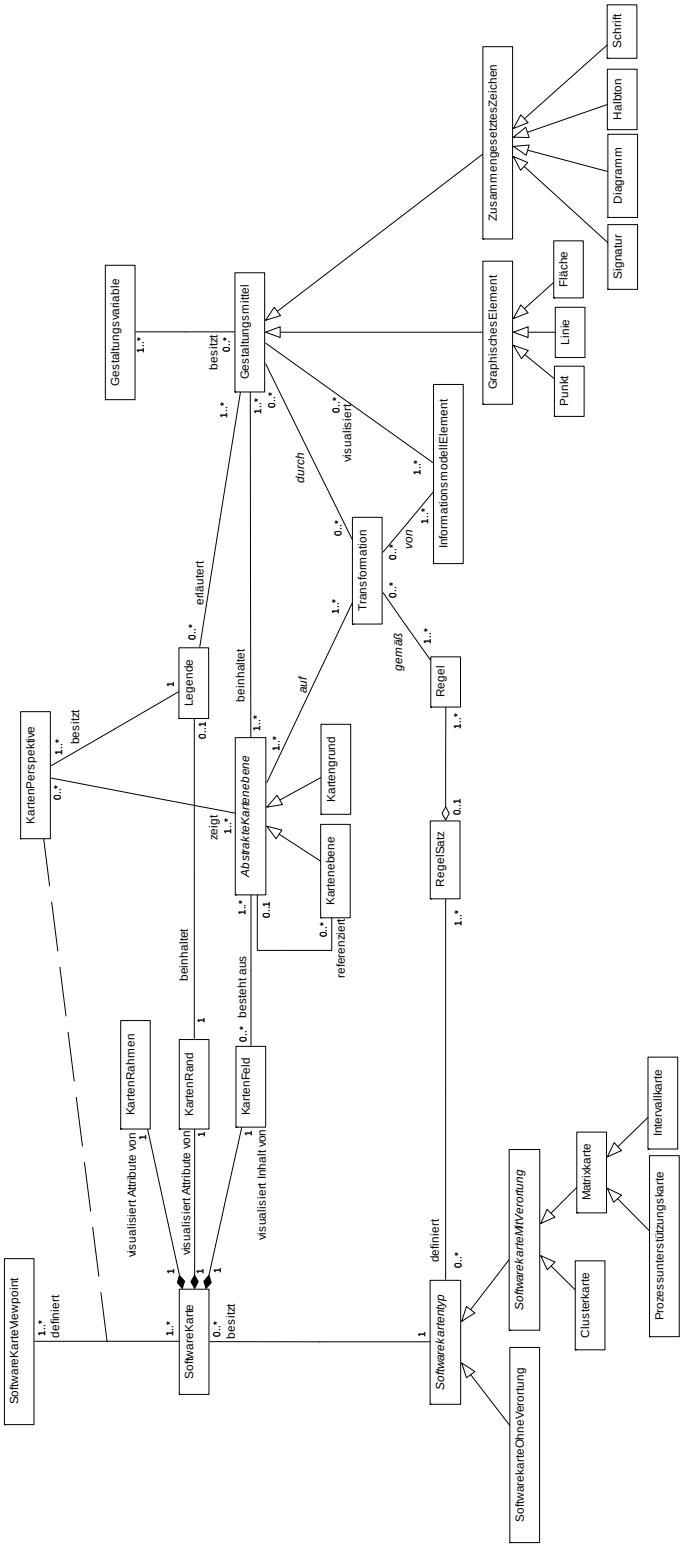


Abbildung 19: Konzeptuelles Visualisierungsmodell

4.4 Zusammenfassung der Analyseergebnisse

Die Analyse bestehender Diagramme und Karten der HVB Systems bestätigt den Bedarf an unterschiedlichen Sichten der Unternehmensarchitektur, die in Form von Softwarekarten visualisiert werden. Die Mitarbeit an aktuellen Themen der HVB Systems Architekturabteilung machte deutlich, dass eine Sicht auf die Geschäftsprozesse des Unternehmens und den sie unterstützenden Anwendungssystemen derzeit von großer Bedeutung ist. Die Visualisierung wie sie in der Building Block Landkarte (siehe Anhang A) oder im IT-Bebauungsplan vorzufinden ist, findet in ähnlichen Ausprägungen häufig Verwendung, um unter anderem die IT-Strategie des Unternehmens zu steuern.

Neben der Sicht auf die Anwendungslandschaft zeigte sich jedoch auch die Notwendigkeit einer geeigneten Repräsentation der Softwarearchitekturen (siehe Analyse des Softwarearchitektur-Bebauungsplanes Anhang A). Eine Sicht, die die Struktur eines Anwendungssystems in Form von Schichten und Bausteinen bis hin zu konkreten Produkten veranschaulicht, liefert das notwendige Verständnis über den Zusammenhang von Anwendungssystemen und den in ihn eingesetzten Produkten.

Die dritte, sich aus der Analyse ergebende Sicht, ist die Sicht auf die technischen Produkte, die innerhalb der HVB Systems für die Entwicklung und den Einsatz von Anwendungssystemen eingesetzt werden. In dieser Sicht stehen die Kenntnis der gültigen Produktkombinationen und die Produktlebenszyklen im Vordergrund.

Es wurde deutlich, dass je nach Sicht einschließlich deren Stakeholder und zu beantwortenden *Questions,* unterschiedliche Darstellungsformen (Softwarekartentypen) und Darstellungsmittel benötigt werden.

Das häufig in der Literatur genannte Problem der verstreuten Informationen, traf auch auf die Informationsbeschaffung der Kartenanalyse zu [BH04]. Die Schwierigkeit lag zunächst darin, genaue Informationen über den Entstehungsprozess jeder einzelnen Karte zu beschaffen. Es konnten keine umfassenden Dokumente gefunden werden, die beginnend bei den Stakeholdern mit ihren *Concerns*, über *Questions* der Analysten bis hin zu den gewählten Gestaltungsprinzipien des *Modelers* erläutern, warum die Karte entstand und warum sie das vorliegende Erscheinungsbild besitzt. Auch die einzelnen Softwarekarten enthalten keine vollständigen Legenden, die alle Kartenelemente erläutern. Allein aus Gesprächen mit Mitarbeitern, Microsoft Powerpoint Präsentationen zu Karten-Projektarbeiten und einem "Karten-Reegineering" wurden die aufgeführten Informationen zusammengestellt.

Bei der Untersuchung der Rollen ließ sich feststellen, dass oftmals mehrere Rollen von gleichen Personen eingenommen werden. Bei den Technologieset-Karten werden beispielsweise die drei Rollen Stakeholder, Analyst und *Modeler* vom Architekten eingenommen. Oftmals gibt es bei der Kartenerstellung ein Projektteam, das aus den genannten Rollen besteht, wo jedoch keine klare Trennung der Rollen vorgenommen werden kann. Des Weiteren ist die Belegung der Rollen oft Unternehmensspezifisch und somit abhängig von der Unternehmensorganisation.

Für die Angabe der Anzahl an Personen einer Rolle zeigte sich, dass sie insbesondere für die Kartenaktualisierung von Bedeutung ist. Handelt es sich z.B. um mehr als 100 *Information Supplier* und soll die Kartenerneuerung laut Vorgabe innerhalb einer Woche abgeschlossen werden, so lässt sich bereits im Vorfeld erkennen, dass eventuell der Zeitraum für die Pflege niemals eingehalten werden kann bzw. ein geeignetes Vorgehensmodell zu entwickeln ist, dass streng durchgezogen werden muss.

Bei genauer Betrachtung der eingesetzten Gestaltungsprinzipien wird deutlich, dass sich derzeit nur wenig Erkenntnisse und Prinzipien der Kartographie in der Modellierung der Softwarekarte wieder finden lassen. Erste Ansätze, wie z.B. die Wahl von aussagekräftigen symbolischen Signaturen oder die Farbwahl nach dem Ampelprinzip, lassen sich erkennen, doch scheinen die Prinzipien der Kartographie nicht ausgeschöpft zu werden. Ein Defizit liegt im Gebrauch von Farbcodes. Die Farbwahl für Kartenelemente erhält oftmals keine Bedeutung, sondern wird rein subjektiv festgelegt (vgl. Rahmenfarbe der IT-Szenarien im IT-Bebauungsplan). Zudem müssen Farben insbesondere bei Online Repräsentationen nach Kriterien der Corporate Identity gewählt werden, so dass ein Farbcode mit einer besonderen Bedeutung für die Kartenelemente nur eingeschränkt eingesetzt werden kann.

Zudem ist die Wahl von symbolischen Signaturen suboptimal. In Gesprächen mit Kartennutzern wurde deutlich, dass die Signaturen teilweise den Sinn der nutzenbringenden Visualisierung von Informationen verfehlen. So ist z.B. die Signatur für die Produkte und Produktversionen in den Technologiesets nicht geeignet, da die wesentliche Information, der Produktnamen, nur schlecht zu lesen ist. Eine große Technologieset-Karte scheint bei einer Betrachtung aus der Ferne nur aus grünen P's zu bestehen, diese Information ist jedoch für den Kartennutzer nicht hilfreich. Erst bei einem Zoom-In werden die wesentlichen Elemente, die Namen der Produkte und ihrer Versionen, lesbar.

Eine zweite Problematik bei der Verwendung von symbolischen Signaturen liegt darin, dass Karten teilweise gleiche symbolische Signaturen enthalten, obwohl sich die Bedeutung dieser unterscheidet. Ein Beispiel ist die symbolische Signatur für Anwendungssysteme im IT-Bebauungsplan. Diese Signatur stammt ursprünglich aus einer anderen Softwarekarte, bei der der äußere Rahmen eine spezielle Bedeutung hatte. Zunächst scheint es von Vorteil zu sein, eine bereits eingesetzte Signatur für das gleiche darzustellende Objekt wieder zu verwenden, allerdings kann dies auch zu Problemen führen: die Signatur ist angesichts ihres Informationsgehaltes viel zu komplex und der Kartennutzer ist aufgrund der unterschiedlichen Interpretationen irritiert.

Ein weiteres Defizit liegt in den aktuell eingesetzten Modellierungswerkzeugen. Sie unterstützen weder das nach Lankes et al. [LMW05a] beschriebene Schichtenprinzip, noch die Wiederverwendung von Modellen bzw. Kartengründen in neuen *Viewpoints*. Folglich ergeben sich derzeit Karten, die gleiche Informationsobjekte enthalten, sich aber in ihrer Darstellung unterscheiden, weil sie unabhängig voneinander entwickelt werden. Ein Beispiel sind der IT-Bebauungsplan und die Building Block Landkarte. Beide Karten sind Prozessunterstützungskarten und liefern Informationen über den Zusammenhang von Prozessen und Anwendungssystemen. Allein die Tatsache, dass der IT-Bebauungsplan projektspezifisch ist und die Building Block Landkarte die Beziehung zu den Building Blocks beinhaltet, reicht aus, dass zwei völlig unterschiedlichen Karten entstehen. Obgleich sich bei genauer Betrachtung feststellen lässt, dass beide Karten auf den gleichen Kartengrund aufgetragen werden könnten, fehlt die Möglichkeit einer selektiven Informationsrepräsentation mittels des Schichtenprinzips. Mit Hilfe eines geeigneten Softwarekarten Werkzeuges ließe sich die Anzahl unterschiedlicher Kartendarstellungen reduzieren, so dass Kartennutzer auf gewohnte Symboliken treffen und sich in neuen *Viewpoints* schneller zurechtfinden würden.

Wie Matthes et al. [MW04a] beschreiben und bereits in vorangehenden Abschnitten festgestellt wurde, sind ein geeignetes Repository, ein Prozess zur steten Aktualisierung der Daten und eine (Semi-) Automatisierung des Visualisierungsprozesse unumgänglich, wenn die Softwarekarten als Steuerungsinstrument für Anwendungslandschaften genutzt werden und einen dauerhaften Wertbeitrag liefern sollen. Bei obigen Karten dient zwar teilweise das Firmen Repository als Datenlieferant für den *Information Supplierer*, doch werden die Daten überwiegend per Hand auf den Karten platziert, da nicht alle benötigten Informationen im Repository vorhanden sind. Speziell die Informationen über Zusammenhänge verschiedener Objekte fehlen. Die Frage welches Anwendungssystem welchen Prozess unterstützt, kann mit Hilfe des derzeitigen Repositorys nicht beantwortet werden. Gemäß der Aussage: „im Code liegt die Wahrheit“, wäre für die Beschaffung aller benötigten Informationen der Ansatz der Introspektion sehr interessant. Das Ziel besteht darin, dass der Programmcode eines Anwendungssystems introspektiv über seine Eigenschaften und Zusammenhänge zu anderen Objekten Auskunft gibt (vgl. [Bü05][20]).

Ein Aspekt, der bislang in der Softwarekartographie noch nicht näher untersucht wurde, ist die Frage nach dem Umgang mit Kartenausschnitten. Oftmals wird die Größe einer Softwarekarte durch anwendungsspezifische Anforderungen festgelegt. Beispielsweise darf eine Softwarekarte nicht Größer als ein DIN A4 Blatt (vgl. Building Block Landkarte 91) sein oder bei einer bestimmten Bildschirmauflösung und -größe die Bildschirmränder nicht überschreiten. Die eine Möglichkeit derartigen Anforderungen gerecht zu werden ist per Zoom-Out die Größe der Karte auf die gewünschte Größe zu reduzieren. Allerdings liefert dieses Verfahren nicht immer die gewünschte Ausgabe, denn durch das Verkleinern der Karte werden einige Informationen unlesbar oder werden gänzlich gelöscht (siehe Abbildung 20).

[20] Forschungsprojekt „Introspektion“ des sebis Lehrstuhls.

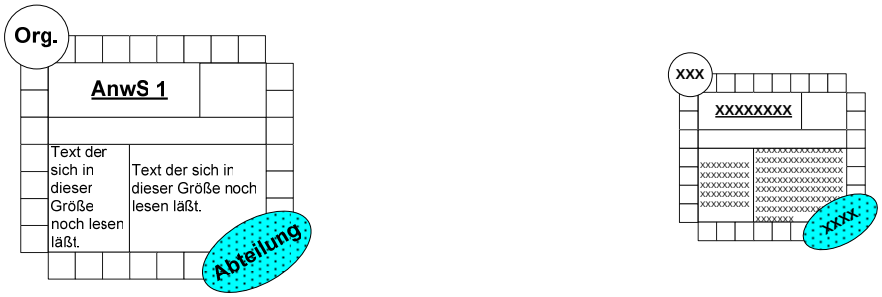


Abbildung 20: Informationsverlust bei einem Zoom-Out über Prozentangaben in Visio[21]

Aus diesem Grund ist eine geeignete Methode zu entwickeln, die zum einen die Karte in einzelne Kartenausschnitte gewünschter Größe zerlegt und zum anderen den Zusammenhang zwischen den Kartenausschnitte für den Kartennutzer verdeutlicht. Der Nutzer muss die Möglichkeit besitzen, über geeignete Funktionen oder Symbole zu den anliegenden Kartenausschnitten zu wechseln.

In einem Beitrag zur Modellierung inter-organisationaler Geschäftsprozesse mit Hilfe von Ereignisgesteuerte Prozessketten [KKS04] wurde auf diese Problematik Bezug genommen. Im Folgenden sollen die zwei dort gewählten Lösungsansätze auf die Softwarekarten übertragen werden.

Der erste Lösungsansatz besteht darin, die Karte "zu zerschneiden". Bei dieser so genannten *horizontalen Zerlegung* bleiben die Kartenelement wie in der ursprünglichen Karte erhalten. Je Ausschnitt werden so viele Kartenelemente angezeigt, wie es die Größe der Fläche zulässt. Alle anderen Elemente werden abgeschnitten. Die Verlinkung zu anschließenden Kartenausschnitten lässt sich über Wegweiser realisieren, die als Sprungmarken zwischen den einzelnen Teilkarten dienen, indem sie die Referenz auf ein Kartenelement der anliegenden Karte beinhalten. Beispielhaft veranschaulicht Abbildung 21 die horizontale Zerlegung, bei der ein Nutzer durch einen Klick auf den Prozesswegweiter (grüner Pfeil) in der linken Teilkarte, zur rechten Teilkarte gelangt. Um dem Benutzer ein besseres Kontextverständnis zu ermöglichen, sollten jeweils die Ausgangs- und die Zielteilkarte die Wegweiser-Elemente als gemeinsame Kartenbestandteile besitzen. In einer online Kartendarstellung könnte die horizontale Zerlegung z.B. mittels Verlinkung gelöst werden, bei ausgedruckten Karten durch Symbolik und geeignete Nummerierung der Kartenausschnitte (ähnlich wie bei Landkarten).

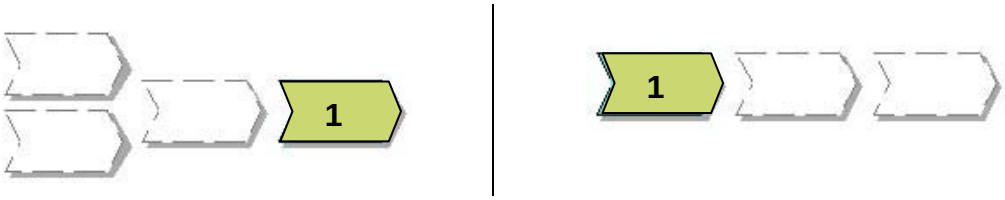


Abbildung 21: Horizontale Zerlegung

Die zweite Möglichkeit stellt eine *hierarchische Zerlegung* dar, bei der mittels semantischer Detaillierung einzelner Kartenelemente das Abstraktionsniveau und somit die Granularität und auch die Größe der Karte beeinflusst werden kann. Bei dieser Variante besteht die Karte aus hierarchisierten Elementen und elementaren Elementen, wobei sich die hierarchisierten Elemente in Teil-hierarchisierte Elemente oder elementare Elemente zerlegen lassen. Abbildung 21 zeigt ein Beispiel für die hierarchische Zerlegung, wobei weiße Elemente für die elementaren Elemente stehen und das grüne das hierarchische Element repräsentiert. Durch einen Klick auf das hierarchische Element in der linken Ansicht (grobgranular), gelangt der Nutzer zur rechten Ansicht (feingranular). Hierarchisierte Elemente müssen demnach über eine geeignete Aufklapp- und Zuklapp- Funktion ihrer Teilbäume verfügen.

[21] Microsoft.

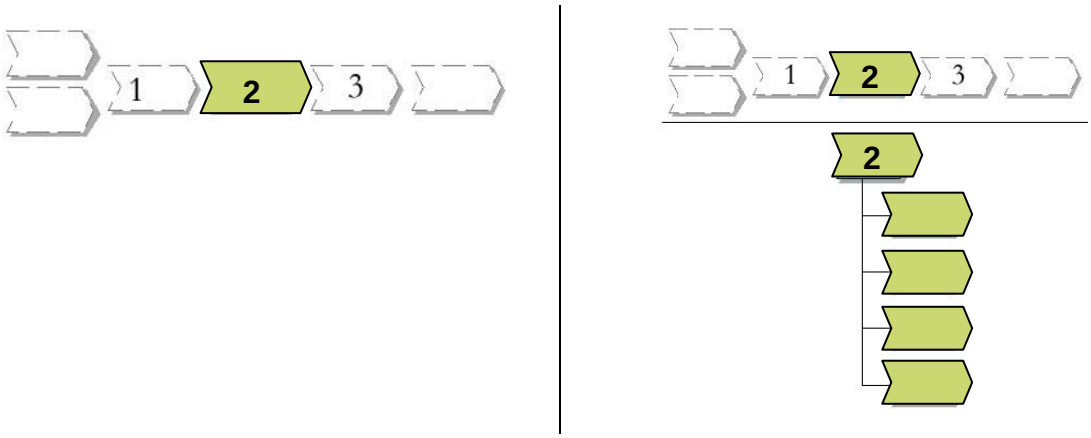


Abbildung 22: Hierarchische Zerlegung

Aus den gewonnenen Erfahrungen lassen sich ergänzend zu den Anforderungen aus den Publikationen des sebis Lehrstuhls [LMW05b, LMW05a, MW04a, MW04b] folgende Vorschläge bzw. Anforderung an das Werkzeug zur Softwarekartographie aufstellen:

- Möglichkeit der Dokumentationserstellung und -verwaltung rund um das Thema Kartenentwicklung und *Viewpoint*: Warum wurde die Softwarekarte entwickelt? Wie kam es zu den gewählten Gestaltungsmitteln? Wann wurde die Karte erstellt?
- Mechanismus bzw. Anreize um den Werkzeugnutzer zur Dokumentation anzuregen: Wie kann man den Werkzeugnutzer von der Notwendigkeit einer Kartendokumentation überzeugen? Wie kann man ihn belohnen? Wie kann man die Dokumentation für den Nutzer erleichtern, so dass die Erstellung für ihn keine besondere Last ist?
- Zwang der Anzeige einer Legende für jede Softwarekarte, in der die Bedeutung jedes Elementes aufgelistet wird: Lässt sich eine Legende soweit generieren, dass der *Modeler* nur die Informationen zu den Symbolen einfügen muss, anstelle die einzelnen Symbole händisch zu einer Legende zusammenzubauen?
- Flexibilität bei der Werkzeugumsetzung des Rollenkonzeptes bestehend aus Stakeholder, Analyst, *Modeler* und *Information Supplier*. Soll diese beispielsweise bei der Rechtevergabe des Werkzeuges zutragen kommen, muss darauf geachtet werden, das es in der Praxis eventuell schwer fällt, Benutzer den definierten Rollen zuzuordnen. Aus diesem Grund sollte das System flexibel genug sein, um neue Rollen definieren zu können oder Rollen unabhängige Rechtevergabe zu ermöglichen.
- Angebot eines Repositorys für Gestaltungsmittel: Bildhafte Signaturen, Symbolische Signaturen, Werteinheitszeichen, Geometrische Signaturen, Lineare Signaturen, Diagramme. Insbesondere soll der Werkzeugnutzer die Möglichkeit besitzen, eigene Signaturen in das Repository einzupflegen und zu verwalten. Diese Funktion soll den *Modeler* bei der Wahl geeigneter Signaturen und die Verwendung gleicher Signaturen mit gleicher Bedeutung unterstützen.
- Rasterausrichtung der Kartenelemente sollte bei Wunsch automatisch erfolgen, falls per Hand Elemente verschoben wurden.
- Bereitstellung von Richtlinien bzw. Gestaltungsprinzipien für Softwarekarten, die sich aus den Erkenntnissen der Kartographie ergeben: Verortung von Elementen, horizontale und vertikale Ausdehnung von Objekten, Farbcodes, Schichtenprinzip, Einsatz von geeigneten Signaturen.
- Hinweise auf weitere Rahmenbedingungen bei der Kartenerstellung: z.B. wurde bereits die Internet Präsenz durch externe Firmen (z.B. Frauenhofer Institut) beurteilt und bestimmte Richtlinien definiert, die unbedingt berücksichtigt werden müssen? Existieren Firmen interne Gestaltungsrichtlinien (Corporate Identity)?
- Einbindung eines Vorgehensmodells für die Kartenerstellung z.B. in Form eines Beispiels oder Negativbeispiels.

- Umsetzung von Methoden zur Veränderung der Kartenausgabegröße z.B. oben genannte horizontale und hierarchische Zerlegung.

Die Zusammenarbeit mit der HVB Systems zeigte einen starken Bedarf eines Softwarekartographie-Werkzeugs. Damit das Werkzeug neben den existierenden Systemen, wie beispielsweise dem Firmen Repository oder ARIS Toolset (IDS Scheer) bestehen kann, sollte es lediglich die Aufgabe der Visualisierung von Sachverhalten in Form der Softwarekarten übernehmen. Eine zusätzliche Methodik bzw. zu umfangreiche Funktionalitäten (Repository, Projektmanagement) würden die politische Entscheidung für bzw. gegen ein solches neues Werkzeug nur negativ beeinträchtigen. Das Softwarekartographie Werkzeug sollte nicht versuchen, mit den bereits in der Firma existierenden Unternehmensarchitektur-Werkzeugen zu konkurrieren, sondern Schnittstellen anbieten, um für die Visualisierung benötigten Daten zu importieren.

Abschließend lässt sich festhalten, dass die Analyse der bestehenden Softwarekarten der HVB Systems, Hinweis auf die ersten drei Sichten und die ersten relevanten Objekte für das Informationsmodell des IT-Managements geliefert hat. Das anschließende Kapitel setzt auf diesen Erkenntnissen auf und umfasst die Entwicklung eines Informationsmodells.

Kapitel 5

Entwicklung eines integrierten Informationsmodells

Das IT-Management verlangt eine integrative Sicht auf die Unternehmensarchitektur, um den Gestaltungsprozess der IT-Landschaft verstehen und geeignet steuern zu können. Im folgenden Kapitel soll durch die Modellierung eines geeigneten Modells, das verschiedene Objekte, wie Prozesse, Organisationseinheiten, Informationssysteme und technische Komponenten, geeignet in Beziehung setzt, das Verständnis, die Dokumentation, Kommunikation und Analyse der IT verbessert werden [Oe03]. Modelladressaten sind somit Stakeholder, die sich einen übergreifenden Überblick über die Zusammenhänge der Unternehmensarchitektur schaffen wollen. Teilmodelle können darüber hinaus für konkrete Umsetzungen in Form von Datenbanken oder Visualisierungen wie beispielsweise die Softwarekarten, benutzt werden, so dass auch Personen, die Informationen über Teilbereiche des Modells benötigen, von diesem Modell profitieren können. Ziel des Informationsmodells ist es, abstrakte Sichten für die Ausrichtung von Geschäft und IT zu liefern, die das Gesamtziel – die Strategien und Ziele von Geschäft und IT zu verwirklichen – im Auge haben (siehe Abschnitt 2.1 und 2.2).

Die Modellierung bedeutungsvoller Objekte, die erfasst und definiert werden, sowie die Diskussion komplexer Sachverhalte auf geeigneter Abstraktionsebene, liefert ein besseres Verständnis der Stakeholder über existierende Gegebenheiten [Sc04b]. Nach Esser [Es02] schaffen Modelle Transparenz über die Elemente und Beziehungen innerhalb des Unternehmens. Das Modell, das auf Papier ausgedruckt oder in elektronischer Form auf einem Datenträger abgespeichert werden kann, liefert eine Dokumentation der beschriebenen Sachverhalte, die für alle Beteiligten als Referenzwerk dient und einen Begriffsapparat liefert. Bernhard et al. [BBB03] stellen sogar fest, „(…) je besser die IT-Organisation dokumentiert ist, desto größer ist die Wahrscheinlichkeit, dass die Leistungsfähigkeit der IT hoch ist“. Zudem eignet sich das Modell, um die Kommunikation im Unternehmen zu unterstützen [Es02]. Es bietet zudem die Basis für eine Analyse der Geschehnisse und Komponenten der IT, da durch die Angabe und Festlegung bestimmter Eigenschaften von Objekten, Beziehungen und deren Kardinalitäten ein Vergleich zwischen Ist- und Sollwerten durchgeführt werden kann. Differenzen und Handlungsbedürfnisse können identifiziert und einer Lösung zugeführt werden.

Im folgenden Abschnitt 5.1 wird zunächst der Begriff des Modells in Bezug auf die Unternehmensarchitektur aufgegriffen, indem einige Modelle aus der Theorie und Praxis eingeführt werden. Die gewonnenen Erfahrungen sollen in einem integrierten Informationsmodell für die Unternehmensarchitektur und somit für das IT-Management[22] zusammengeführt werden, dass den obigen Anforderungen gerecht wird. Abschnitt 5.2 widmet sich der Entwicklung des Informationsmodells und beschreibt seine einzelnen Komponenten, welche in ihrer Gesamtheit in Abschnitt 5.3 anhand der Grundsätze ordnungsmäßiger Modellierung kritisch bewertet werden. Eine Zusammenfassung des Kapitels erfolgt in Abschnitt 5.4.

5.1 Modelle für die Unternehmensarchitektur und dessen Teilbereiche

Wie bereits im einleitenden Kapitel 2 erläutert wurde, soll die Unternehmensarchitektur die Brücke zwischen dem Geschäft und der IT bilden. Ist dabei von dem Management der Unternehmensarchitektur die Sprache, so handelt es sich um den Prozess zur Kontrolle der bestehenden, sowie zur Planung und Durchsetzung einer verbesserten IT-Unterstützung einer Organisation [Seb04]. Dieser kontinuierliche, iterativ ablaufende Prozess befasst sich mit einer großen Anzahl komplexer Komponenten, so dass es sinnvoll ist die Unternehmensarchitektur in funktional zusammengehörige Bereiche zu untergliedern und für jeden Teilbereich Informationen über die Unternehmensarchitektur erhebbar, verfügbar, aggregierbar, filterbar und verarbeitbar bereitzustellen [Seb04].

Abbildung 23 veranschaulicht exemplarisch die Sicht der Siemens AG auf vorhandene Informationsquellen im Unternehmen, welche bereits vom Unternehmensarchitektur-Management genutzt werden. So existieren Informationen rund um die Geschäftsprozesse in der HVB Systems beispielsweise im

[22] Siehe Abschnitt 2.1 und 2.2.

ARIS Toolset (IDS Scheer). Das HVB Systems Repository (basierend auf Rochade) und beispielsweise das Softwaremodellierungswerkzeug Rational Rose (IBM) verwalten die Informationen, die die Anwendungssysteme des Unternehmens betreffen. Projektinformationen können über Anwendungen wie MS Project (Microsoft) oder SAP Anwendungen abgebildet werden. Weitere spezielle Systeme können Angaben zu den die IT bereitstellenden Diensten oder die Beziehungen zwischen Hardware und Anwendungssystemen verwalten.

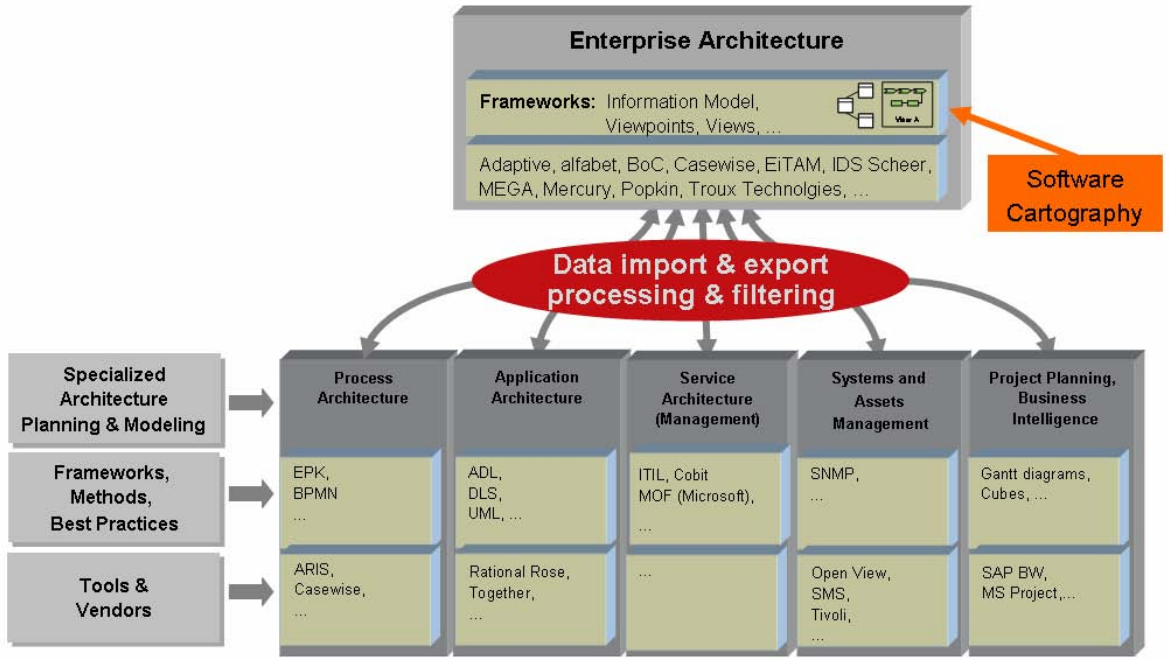


Abbildung 23: Quellen für das Architekturmanagement angelehnt an Siemens CIO 2005

In der Literatur und in der Praxis existieren Modelle, die Organisation und Aufbau von Unternehmensarchitekturen beschreiben und Vorschläge für die Erarbeitung bzw. die Etablierung eines Unternehmensarchitektur-Managementprozesses bieten. Häufig werden die aufgeführten *best-practices* in einem Framework zusammengefasst, dass von Unternehmen als ein Werkzeug verwendet werden kann. Folgend sollen einige Modelle für die Unternehmensarchitektur bzw. Ansätze für Informationsmodelle vorgestellt werden, die zum Entstehungsprozess des in Abschnitt 5.2 eingeführten Modells beitragen.

5.1.1 Zachman Framework

Im Jahr 1987 entwickelte John Zachman, Gründer des *Zachman Institute for Framework Advancement* (ZIFA) ein Rahmenwerk für Informationssystemarchitekturen, dass 1992 in Zusammenarbeit mit Sowa um weitere Aspekte ergänzt wurde. Das Framework beschränkt sich dabei auf die reine Architektur einschließlich dessen Definition bzw. Beschreibung und beinhalten keine strategisch, planende Methodik [Za87].

Nach seiner Einführung gewann das Zachman Framework bei Systemanalysten und Datenbankdesignern mehr und mehr an Akzeptanz [SZ92]. Es kristallisierte sich heraus, dass es sich neben der Beschreibung von Informationssystemarchitekturen auch für den Umgang mit der Unternehmensarchitektur eignet. Das heutige Ziel der ZIFA ist es, das Zachman Framework als universelle Sprache für die Kommunikation, Untersuchung und Implementierung von Konzepten der Unternehmensarchitektur zu etablieren [Za04].

Nach der Definition von Zachman existiert nicht *eine* Informationssystemarchitektur, sondern ein Satz von Architekturen, die sich gegenseitig ergänzen und die Architektur des Systems aus unterschiedlichen Perspektiven durchleuchten [Za87, SZ92].

5.1.1.1 Konzept

Das erweiterte Rahmenwerk besteht aus einer Matrix mit sechs Zeilen und sechs Spalten (siehe Abbildung 24), die die Unternehmensarchitektur in 36 zu beschreibende Bereiche unterteilt. Je Bereich gibt die Spalte relevante Informationsobjekte und die Zeile die Stakeholder inklusive der benötigten Modellierung vor.

Spalten untergliedern die Unternehmensarchitektur anhand verschiedener Level. Der erste Level umfasst die Daten des Unternehmens, die Frage nach dem „was“. In der zweiten Spalte stehen die Funktionen, d.h. das „wie“ im Mittelpunkt. Informationen über den Verbund einzelner bzw. die Lokation beteiligter Objekte, werden in der dritten Spalte des Netzwerks, dem „wo“ gesammelt. Drei weitere Level umfassen Angaben über die beteiligten Personen („wer“), den zeitlichen Zusammenhängen („wann“) und der Motivation („warum“).

Je Level wird zwischen unterschiedlichen Perspektiven unterschieden, die Auskunft über den Abstraktionsgrad und die Art der Modellierung eines jeweiligen Levels geben. Die oberste Perspektive ist die des Planers, der eine kontextabhängige Sicht der unterschiedlichen Level auf das Unternehmen benötigt. Sichten bestehen in dieser Perspektive aus Listen, die Informationen über die Instanzen der Informationsobjekte liefern. Aus der Perspektive des Inhabers, besitzen die Sichten dahingegen konzeptionellen Charakter, in denen die Beziehungen der Informationsobjekte dargestellt werden. Eine Perspektive tiefer benötigt der Designer logische Informationen, die in Form von Systemmodellen zusammengefasst werden. Physikalische Aspekte sind für den Erbauer und in detaillierter Form für den Zulieferer relevant. Die sechste Ebene repräsentiert das funktionsfähige Unternehmen.

5.1.1.2 Fazit

Das Zachman Framework liefert den Hinweis auf das Bewusstsein verschiedener Stakeholder. Das zu entwickelnde Informationsmodell muss die Möglichkeit bieten, Informationen aus Teilbereichen des Gesamtmodells als eigenständige Sichten zur Verfügung zu stellen. Die Einführung mehrerer Perspektiven und Level für unterschiedliche Fragestellungen weißt darüber hinaus auf eine notwendige Strukturierung des Modells in Teilmodelle hin. Einzelne Objekte der 36 Bereiche können als Informationsobjekte für das Modell übernommen werden.

Der Nachteil des Zachman Frameworks liegt jedoch in der fehlenden Integration der unterschiedlichen Bereiche. Obwohl es in der Praxis häufig zitiert wird und als Anregung eingesetzt wird, konnte im Rahmen dieser Arbeit keine konkrete Anwendung gefunden werden, die es in aller Vollständigkeit einsetzt. Auch die Konkurrenz bzw. die Unverträglichkeit zu anderen Modellen wie beispielsweise ARIS Toolset (IDS Scheer) und der Meta Group ist als negativ zu betrachten.

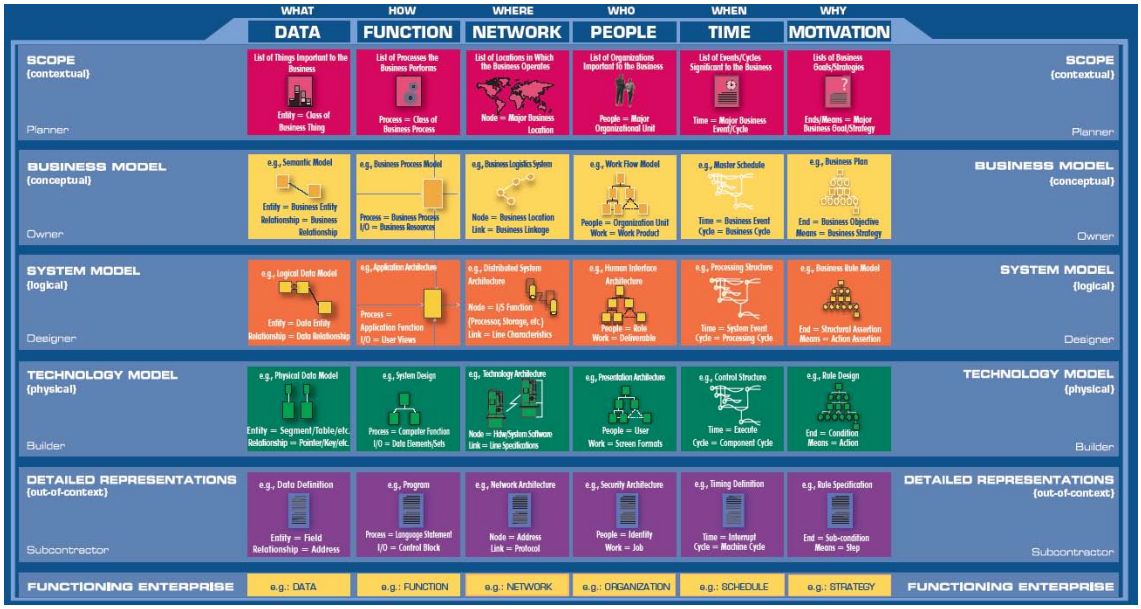


Abbildung 24: Elemente des Zachman Frameworks aus [Za04]

5.1.2 Expanded Gartner Architecture Framework

Die Betrachtungsebene des Gartner Architektur Framework ist der von Zachman sehr ähnlich, weil es ebenfalls Komponenten und Zusammenhänge der Unternehmensarchitektur und dessen Elemente vereinigt. Zachman definiert Spalten und Zeilen, bei Gartner bekommt das Modell aufgrund mehrerer Schichten und Aspekte einen mehrdimensionalen Charakter (vgl. [Sc04a]).

5.1.2.1 Konzept

Die drei Dimensionen des Expanded Gartner Architecture Frameworks sind die *Business Architecture*, die Ebenen für Geschäftsprozesse und weitere geschäftsrelevante Aspekte enthält, die *Technical Architecture* mit der Infrastruktur-, System-Management- und der Security-Domäne und der *Information Architecture* mit den Domänen *Data*, *Application*, *Integration* und *Point of Access* (siehe Abbildung 25).

Das Modell definiert die Bestandteile der jeweiligen Ebenen und ordnet Probleme, die in der Praxis bei der Entwicklung der Unternehmensarchitektur auftreten entsprechenden Ebenen zu. Als besonderer Bestandteil des Modells lassen sich die *Patterns* (deutsch Muster) und *Bricks* (deutsch Bausteine) bezeichnen, die Werkzeuge für den Architekten darstellen. Mit Hilfe dieser Konzepte kann der Architekt Pläne für die Gestaltung der Architektur entwickeln [Sc04a].

5.1.2.2 Fazit

Das Gartner Framework liefert für das zu entwickelnde Informationsmodell außer der Strukturierung und den Objekten *Business Processes*, *Patterns* und *Bricks* keine Hinweise auf relevante Objekte oder Begriffsdefinitionen.

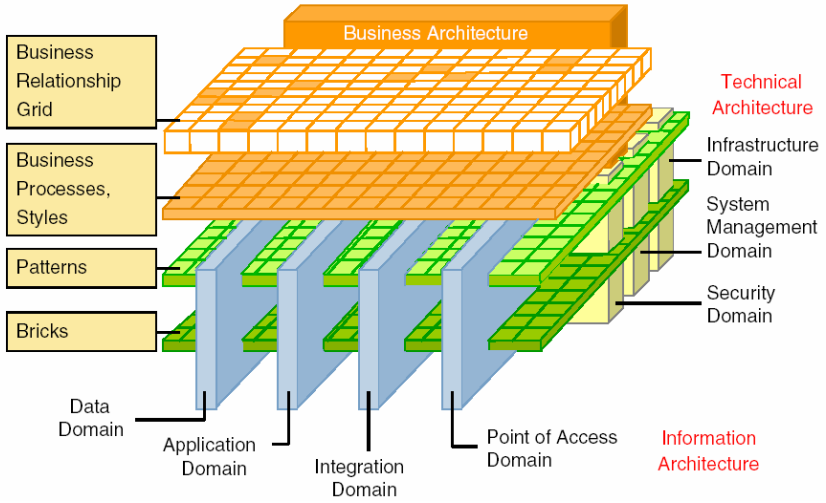


Abbildung 25: Expanded Gartner Architecture Framework aus [Sc04a]

5.1.3 The Open Group Architectural Framework

The Open Group Architectural Framework (TOGAF) ist seit Mitte der neunziger Jahre das Architekturframework der *Open Group* und ermöglicht das Design, die Evaluierung und Erstellung der „richtigen“ Architektur einer Organisation. Es umfasst als Hauptkern die *Architecture Development Method* (ADM) und stellt Werkzeuge für die Entwicklung der Unternehmensarchitektur bereit [TOG05].

Grundlage des Frameworks bildet das Verständnis der vier zu unterscheidenden Architekturtypen

1. *Geschäfts-Architektur*: Definiert die Geschäftsstrategie, Steuerung, Organisation und die Geschäftsprozesse.
2. *Anwendungssystem-Architektur*: Stellt einen *Blueprint* für individuelle, einzusetzende Anwendungssysteme einschließlich ihren Wechselwirkungen und Beziehungen zu den Kerngeschäftsprozessen der Organisation dar.

3. *Daten-Architektur*: Beschreibt die Struktur der logischen und physischen Datenressourcen.
4. *Technologie-Architektur*: Umfasst die Software Infrastruktur für den Support und die Bereitstellung von Anwendungssystemen.

5.1.3.1 Konzept

Die drei wesentlichen Bestandteile des TOGAF Frameworks sind die bereits genannte ADM, das *Enterprise Continuum* und die TOGAF *Resource Base*. Die *ADM* definiert, wie eine Unternehmensarchitektur unternehmenspezifisch aufgebaut werden kann, indem es ein Vorgehensmodell, mehrere definierte „Architecture Views" sowie verschiedene Richtlinien und Anforderungen für eine Werkzeugunterstützung bereitstellt. Abbildung 26 gibt einen Einblick in das iterative Vorgehen, in dessen Phasen jeweils Inputs, Outputs und Schritte definiert werden. Unter dem *Enterprise Continuum* ist ein virtuelles Repository mit allen Architektur-Hilfen (Modelle, Muster, Architekturbeschreibungen etc.) zu verstehen. Weitere Richtlinien, Vorlagen und Hintergrundinformationen werden von der *Resource Base* zusammengefasst und bereitgestellt.

5.1.3.2 Fazit

TOGAF beschreibt ein generisches Vorgehen und setzt mehrere Perspektiven für unterschiedliche Fragestellungen ein. Die Unterteilung des Begriffs Architektur in die vier oben genannten Typen soll im zu entwickelten Informationsmodell berücksichtigt werden. Da jedoch das zu entwickelnde Informationsmodell eine statische Sicht auf die Unternehmensarchitektur bildet, sind die Konzepte des TOGAF Vorgehensmethodik nicht relevant.

TOGAF konkurriert mit anderen Modellen, das Zachman Framework ausgenommen[23], und beinhaltet keine konkreten Beispiele.

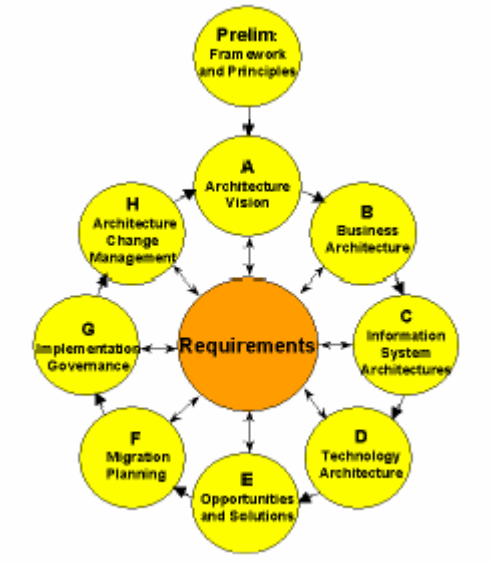


Abbildung 26: TOGAF ADM aus [TOG05]

5.1.4 Common Information Model

Das *Common Information Model* (CIM) wurde 1996 von der *Distributed Management Task Force* (DMTF), einem Industriekonsortium, entwickelt. Es handelt sich um ein objektorientiertes Informationsmodell, das unabhängig von einer konkreten Implementierung, die managementrelevanten Informationen unterschiedlicher Bereiche, u.a. durch grafische Darstellungen beschreibt. Der CIM Ansatz

[23] Eine Integration von TOGAF und Zachman wird in [TOG05] vorgenommen.

stellt verschiedene System-Management-Disziplinen, wie Konfiguration, Sicherheit, Benutzer, Anwendungen, Netzwerke etc., auf eine gemeinsame konzeptionelle Basis dar.

Das Informationsmodell beinhaltet ein Metaschema, das auf der UML basiert und Begriffe wie *Class*, *Property*, *Association* oder *Qualifier* einführt und festlegt. Ergänzend existiert eine textuelle Beschreibungssprache, das *Managed Object Format* (MOF), das auf der *Distributed Computing Environment* (DCE) *Interface Definition Language* (IDL) basiert [DMT05].

5.1.4.1 Konzept

Die in CIM modellierte Information ist in drei Schichten aufgeteilt: dem *Core Model*, dem *Common Model* und dem *Extension Schema*.

Das *Core Model* beschreibt die Informationen, die in vielen Managementbereichen gleichermaßen benötigt werden. Eine erste Repräsentation managebarer Komponenten und eine klare Trennung zwischen Hardware und Softwarekomponenten werden vorgenommen. Beispiele hierfür sind *ManagedElement*, *Product*, *Physical* oder *LogicalElement* sowie die Abhängigkeiten zwischen den Entitäten. Es dient somit als Ausgangspunkt für jegliche Untersuchung und Festlegung von Managementinformationen.

Die *Common Models* erweitern die Information des *Core Models* um Informationen spezifischer Bereiche. Die derzeit modellierten Bereiche sind System, *Application*, *Network*, *Device*, *Physical*, *Events*, *Metrics*, *Policy* und *Support*. Das *Application Modell* beschreibt beispielsweise Klassen die es erlauben, Softwareprodukte in Teilsysteme aufzugliedern, Pakete zu definieren und gegenseitige Abhängigkeiten zu definieren.

Im Rahmen des *Extension Schemas* kann die Beschreibung durch Technologie spezifische Verfeinerung der Information ergänzt werden. Diese Schemata werden in speziellen Umgebungen eingesetzt [DMT05].

5.1.4.2 Fazit

Das CIM Modell eignet sich insbesondere sehr gut für die Ebene der Infrastrukturkomponenten. Auf höheren Ebenen, wie beispielsweise den Geschäftsprozessen oder den Geschäftsstrategien, werden keine Objekte modelliert. Das zu entwickelnde Informationsmodell hat demzufolge einen etwas höheren Abstraktionsgrad, so dass das CIM Modell nur an einigen Stellen für Anregungen genutzt werden kann.

5.1.5 IT-Portfolio Management Facility

ITPMF ist die Abkürzung für die *IT-Portfolio Management Facility Specification*. Hinter dem Begriff IT-Portfolio verbirgt sich zunächst die Gesamtheit der Informationssysteme eines Unternehmens, d.h. soziotechnische Systeme mit dem Ziel der optimalen Bereitstellung von Information und Kommunikation [WKW94], einschließlich der Projekte zu deren Weiterentwicklung über einen mittelfristigen Planungshorizont (5-10 Jahre). Das IT-Portfoliomanagement ist demgemäß ein strukturierter Management Ansatz, der das IT-Portfolio effektiv und effizient an den Unternehmenszielen, den Anforderungen der Geschäftsprozesse und den gesetzlichen Anforderungen ausrichtet. Dieser Ansatz bedarf einer geeigneten Methodik bzw. eines Metamodells, um alle einzusetzenden Software- und Plattformelemente abzubilden und dadurch zu steuern. Dieser Aufgabe hat sich die ITPMF Spezifikation, die derzeit von der OMG standardisiert wird, gewidmet [OMG04].

5.1.5.1 Konzept

Das Metamodell der ITPMF Spezifikation besteht aus einer Menge UML Diagramme, die sich jeweils Teilaspekten des zu modellierenden Sachverhaltes widmen. Im *Essential Diagramm* werden alle Schlüsselklassen der Diagramme zusammengeführt, um einen geeigneten Überblick der einzelnen Diagramme und deren Zusammenhänge zu erhalten.

Kern des Metamodells ist das so genannte *ManagedElement* (oben genannte Software- und Plattformelemente), dessen Eigenschaften und Beziehungen definiert werden sollen. Das *Control Diagramm* umfasst beispielsweise Aspekte rund um die Werte und Lebenszyklen-Stati, die ein *Manage-*

dElement in Abhängigkeit bestimmter Ereignisse einnehmen kann. Im *Context Diagramm* werden die Verträge einschließlich beteiligter Parteien behandelt, die in Bezug auf das *ManagedElement* geschlossen werden können. Handelt es sich bei dem *ManagedElement* um ein Element, dass „deployed“ werden kann, so sind weitere Klassifizierungen dieses *DeployableElements* notwendig. Diese Spezialisierungen werden im *Deployable Elements Diagramm* veranschaulicht. Im *Deployment Diagramm* wird schließlich der Einsatz eines Elementes auf einer Hardware, an einem bestimmten Ort beschrieben.

5.1.5.2 Fazit

Der Fokus des ITPMF Metamodells liegt im Umgang mit den eingesetzten Software- und Plattform Elementen eines Unternehmens. Genannte Aspekte werden bei der Entwicklung der Betriebs-Ebene des zu entwickelnden Informationsmodells berücksichtig (siehe 5.2.1.6).

5.1.6 Zusammenfassung der eingeführten Modelle

Die Vorstellung der Modelle zeigte, dass bereits einige, unterschiedlich gewichtete Ansätze für die Gestaltung bzw. Modellierung der Unternehmensarchitektur und ihren relevanten Objekten existieren. Es wurde deutlich, dass keines der Modelle den Anforderungen des Informationsmodells für das IT-Management gerecht wurde und auch nicht durch einige Veränderungen und Ergänzungen angepasst werden kann.

Aufgabe des nächsten Abschnitts besteht demzufolge darin, ein integriertes Informationsmodell zu entwickeln, das auf geeignetem Abstraktionsgrad, relevante Objekte für die Unternehmensarchitektur und dessen Management benennt, sowie deren Zusammenhänge verdeutlicht. Konzepte die bereits in oben aufgeführten Modellen zu tragen kamen, sollen bei Eignung im Modell angewendet werden.

An dieser Stelle lässt sich bereits die Notwendigkeit einer geeigneten Strukturierung des Modells aus den vorgestellten Modellen ableiten, die darüber hinaus unterschiedliche Sichten des Modells ermöglichen muss, um verschiedenen Stakeholdern und deren Interessen gerecht zu werden.

5.2 Informationsmodell für das IT-Management

Um die Übersichtlichkeit und die Lesbarkeit des Informationsmodells zu verbessern, werden die zu modellierenden Sachverhalte entsprechend einer geeigneten Strukturierung in einzelne Teilmodelle unterteilt. Um dennoch die gegenseitigen Abhängigkeiten und Beziehungen der Teilmodelle zu berücksichtigen, werden die entsprechenden Informationsobjekte in anderen Teilmodellen importiert[24] und dort in Beziehung gesetzt oder als Datentyp eines Attributes verwendet. Der Vorgang der Zerlegung und anschließender Veranschaulichung der Zusammenhänge findet in der Praxis häufig Verwendung und wurde bereits im vorhergehenden Abschnitt anhand einiger Beispiele veranschaulicht.

In Abbildung 27 ist die Paketstruktur des Informationsmodells abgebildet. Auf oberster Ebene besteht das Modell aus dem *IT-Management, IT- und Geschäftsarchitektur* (kurz Architektur) und den Paket sowie der Klasse *Geschäftsstrategie*. Letztere wurde bewusst als einzige Klasse auf Paketebene angehoben, um die Wichtigkeit dieser Klasse zu betonen, sowie dessen Einfluss auf die gesamte Architektur (vgl. [SC04b]).

Laut Gartner [La05a] soll bei der Entwicklung einer Unternehmensarchitektur mit der Geschäftstrategie begonnen und anschließend der Weg zur Technologie fortgesetzt werden, d.h. ein *top-down* Verfahren. Wird eine Unternehmensarchitektur dahingegen durch ein b*ottom-up* Verfahren erarbeitet, kann eine Architektur entstehen, die zwar optimale Technologielösungen definiert, jedoch nicht die Verbindung zwischen Geschäft und IT schafft. Definierte Technologiestandards hätten keine Bedeutung für das Unternehmensgeschäft und keine Verbindung zu Geschäftsfunktionen. Dieser starke gegenseitige Einfluss wird im Modell durch die Beziehungen der Klasse Geschäftsstrategie mit den beiden Paketen IT-Management und Architektur veranschaulicht. Die Notation mittels der <<import>> Beziehung stellt die UML konforme Schreibweise dar und bedeutet, dass in den Paketen IT-

[24] In der UML Notation wird der Import von Klassen durch die zugehörige Paketbezeichnung unterhalb des Klassennamens veranschaulicht (siehe 3.2.2).

Management und Architektur die Klasse Geschäftsarchitektur importiert, d.h. verwendet wird und somit implizit auch auf die restlichen Objekte, wie beispielsweise die Organisationsstruktur, die Softwaresysteme und der Infrastruktur (vgl. [Bo05, BBB03]), einwirkt.

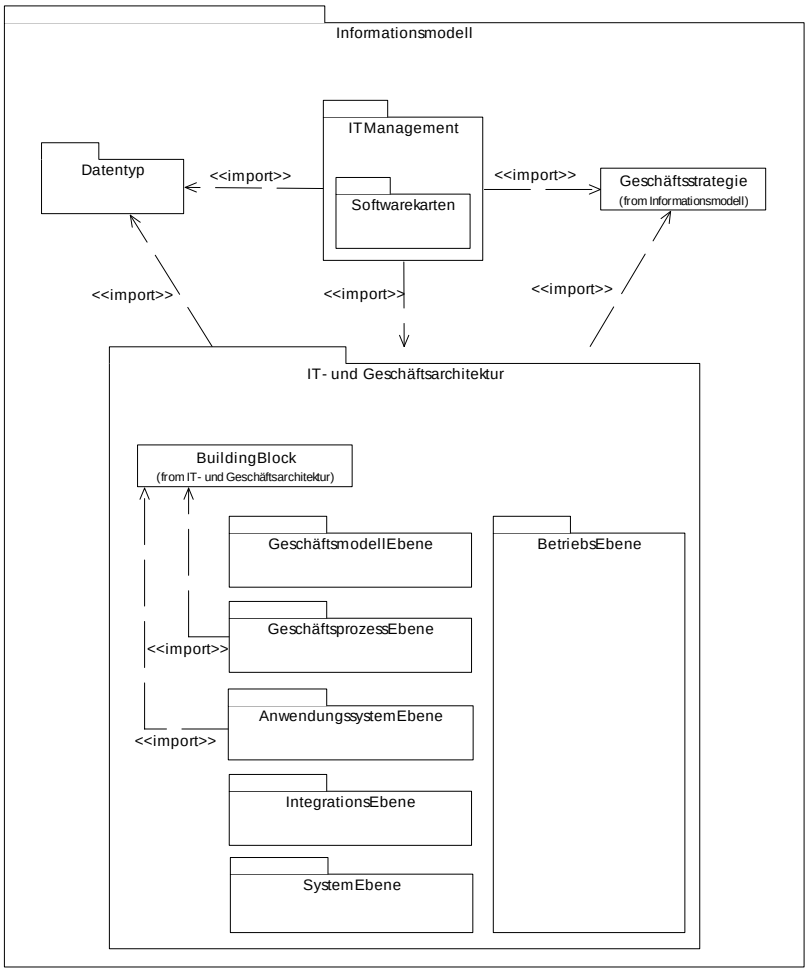


Abbildung 27: Paketstruktur des Informationsmodells

Den Hauptteil des Informationsmodells liefert das Paket *IT- und Geschäftsarchitektur*, da in ihm alle die IT betreffende Informationsobjekte organisatorischer als auch technischer Art enthalten sind. Relevant bedeutet in diesem Zusammenhang, dass die Analyse und das Design nur auf diejenigen Objekte beschränkt werden, die für den Zielsetzungs- und Realisierungsprozess des IT-Managements von Bedeutung sind (vgl. Abschnitt 5.3, Vorgehensweise für Bewertung von relevanten Objekten). Zugehörige Informationsobjekte sind beispielsweise Anwendungssysteme, Produkte und Organisationseinheiten. Die Import-Beziehungen, die zwischen den Teilmodellen des Architektur-Paketes bestehen, wurden aus Übersichtlichkeit nicht dargestellt (vgl. Abbildung 39). Es lässt sich jedoch bemerken, dass kein Paket für sich allein steht sondern mindestens eine Beziehung zu einem anderen Paket existiert. Des Weiteren besitzt das Paket der Betriebs-Ebene einen besonderen Charakter, da es Objekte der anderen Architektur-Teilpakete auf einem anderen Abstraktionsgrad miteinander in Beziehung setzt. Aus diesem Grund wurde es neben den restlichen Paketen als ein übergreifendes Paket dargestellt (vgl. Abbildung 27).

Das *IT-Management*-Paket ergänzt die Objekte des Architektur-Paketes um weitere, das Management betreffende Objekte wie beispielsweise IT-Projekt, IT-Entscheidung oder die Klasse Kosten. Darüber

hinaus beinhaltet es im Teilpaket *Softwarekarten* alle Modelle, die die Informationsobjekte der analysierten Softwarekarten aus Abschnitt 4.2 in Beziehung setzten.

Bevor anschließend die einzelnen Pakete und deren Teilmodelle vorgestellt werden, sollen zunächst einige Hilfsmethoden, die bei der Modellierung angewendet wurden, aufgelistet werden:

1. Beziehungen zwischen Klassen wurden in speziellen Fällen nicht mittels Assoziationen sondern aus Gründen der Übersicht indirekt über Attribute modelliert. Ist beispielsweise ein Anwendungssystem stets einer Organisationseinheit zugeordnet, so wir dies mittels des Attributs Bereich vom Typ Organisationseinheit modelliert. Da bei der Abbildung der Teilmodelle aus Platzgründen auf die Anzeige der Attribute verzichtet wurde, wird an dieser Stellen auf das originale Rational Rose (IBM) Dokument auf der CD in der Anlage dieser Arbeit verwiesen.
2. Attribute werden für eine Klasse nur dann im Informationsmodell aufgenommen, falls es für alle Instanzen mit akzeptablem Aufwand, korrekt erhoben werden kann. In einzelnen Anwendungsgebieten bzw. Fachbereichen[25] können für Objekte durchaus weitere Attribute von Interesse sein. Da diese jedoch meistens sehr speziell sind, wurde im vorliegenden Informationsmodell mittels Interviews der Stakeholder eine geeignete Obermenge von Attributen ausgewählt (vgl. Abschnitt 5.3).
3. Attribute vom Typ Rolle implizieren die Besetzung der Rolle durch eine konkrete Person.
4. Attribute sind vom Typ Kennzahl, wenn durch dessen Ausprägungen die Realität abgebildet wird und die zugehörigen Objekte, beispielsweise Organisationseinheiten oder Anwendungssysteme, mit Hilfe dieses Attributes bzw. dieser Kennzahl gesteuert werden können. Kennzahlen erfassen Sachverhalte quantitativ und in konzentrierter Form [Kü03] und eignen sich somit für eine geeignete Visualisierung auf Softwarekarten, um eine Steuerung der Anwendungslandschaft zu ermöglichen. Die Darstellung von Kennzahlen soll den Informationsgehalt erhöhen und die Eignung von Softwarekarten für Planungs- und Steuerungsaktivitäten verbessern. Da die Herausforderung darin besteht, für eine konkrete Steuerungsaufgabe relevante Informationen zu identifizieren, wurden nur solche Attribute als Kennzahl deklariert, für die ein Visualisierungsszenario vorstellbar wäre. Beispielsweise besitzt ein Anwendungssystem die Kennzahl Risikoklasse und eine Produktversion einen Status, dessen Toleranzwerte in der Technologieset Visualisierung mittels des Ampelprinzips visualisiert werden können (siehe Technologiesets in Anhang A.3). Dieses Attribut könnte bei der Ablösung von Anwendungssystemen Informationen über die damit verbundenen Aufwände transportieren. Die Ablösung von Anwendungssystemen einer niedrigen Risikoklasse rufen sicherlich weniger Aufwände und Kosten hervor, als solche, die in hohen Risikoklassen eingestuft wurden. Für weitergehende Informationen, z.B. welche Kennzahlen von großer Relevanz sind, sei an dieser Stelle auf Beyer [Be04] verweisen.
5. Kardinalitäten mussten teilweise möglichst allgemein definiert werden, indem sie beispielsweise mit einer Null beginnen. Der Grund liegt darin, dass zum einen zwischen obligatorisch und optional zu erfassenden Informationen unterschieden werden muss und zum anderen sowohl die Ist- als auch die Soll-Landschaft zu berücksichtigen ist. Muss ein Anwendungssystem beispielsweise zukünftig immer einem Building Block zugeordnet werden, muss die Kardinalität seitens des Building Blocks dennoch mit einer null beginnen um auch solche Anwendungssysteme aufnehmen zu können, die bislang nicht zugeordnet werden konnten.

Folgende Abschnitte gehen auf die einzelnen Teilmodelle der Pakete ein, wobei jeweils zusammenfassend die wichtigsten Informationsobjekte und Konzepte eingeführt werden. Wurden bei der Modellierung bereits in der Praxis oder der Theorie auftauchende Konzepte verwendet, so werden sie an entsprechender Stelle genannt. Anhang C beinhaltet für jedes Informationsobjekt des Modells eine kurze Definition.

[25] Strukturierung eines Unternehmens in Fachbereich (Marketing, Vertrieb, Produktion, Service einschließlich Akteuren, Daten, Prozesse) und IT-Bereich (IT-Systeme sowie deren Management), siehe Abbildung 64 und Abbildung 65.

5.2.1 IT- und Geschäftsarchitektur-Paket

Die Strukturierung des Paketes über die Architektur erfolgt nach dem IT-Architektur Referenzmodell der HVB Systems, das die Sachverhalte der Unternehmensarchitektur, ähnlich wie die in Abschnitt 5.1 eingeführten Modelle von Zachman und Gartner, strukturiert.

Das Referenzmodell und somit auch das Paket bestehen aus sechs Ebenen, die die IT mit Hilfe unterschiedlicher Sichten von der abstrakten Geschäftswelt bis hin zu konkreten Produkten strukturiert darstellt [HVB03a, Ju04]. Jedes Teilpaket, jede Ebene, ist klar gegeneinander abgegrenzt, verfügt über spezifische Eigenschaften und Funktionen und zeigt zusätzlich durch ein oder mehrere Objekte Schnittstellen bzw. Abhängigkeiten zu anderen Paketen.

Ergänzend zu der Ebenenstruktur, werden gleichartige Funktionsgruppen in so genannten Building Blocks zusammengefasst, die sehr grobgranulare Strukturierungselemente wie z.B. die Wertpapierabwicklung bilden. Ziel ist eine fachlich orientierte Modularisierung nach den Kriterien der Geschäftsstrategie, auf Grundlage der Geschäftsmodell-, Geschäftsprozess-, Anwendungs- und Integrations-Ebene [Ju04].

5.2.1.1 Paket der Geschäftsmodell-Ebene

Die oberste Ebene des IT-Architektur-Referenzmodells stellt die Geschäftsmodell-Ebene dar. Hier wird untersucht und beschrieben, woraus das definierte Geschäft seinen nachhaltigen Erfolg zieht. Kundengruppe, Märkte und die dazugehörigen Produkte der einzelnen Geschäftsfelder werden darlegt [HVB03a].

Zentraler Aspekt des Teilmodells aus dem Geschäftsmodell-Paket ist das Unternehmen selbst mit seinen Unternehmenszielen, aus denen geeignete Strategien abgeleitet werden (siehe Abbildung 28). Die Strategien werden dabei an den Kunden und den Märkten ausgerichtet, so dass das Unternehmen optimale Leistungen für den Kunden bereitstellen kann. Für einen Finanzdienstleister stellen die so genannten Bankprodukte (z.B. Kredit, Immobilien) die entscheidenden Dienstleistungen des Unternehmens dar.

Kunden spielen für ein Dienstleistungsunternehmen eine entscheidende Rolle, da sie der treibende Faktor des Geschäfts und der Geschäftstrategie sind und das Unternehmen mit seiner Umwelt verbinden (vgl. [BBB03, Mü98]). Aus Kundenanforderungen resultieren die Leistungen, die von einem Unternehmen erbracht werden. Bei der Modellierung stellte sich die Frage, wie der Kunde in das Modell integriert werden kann. Das Ergebnis einiger Modellierungsvariationen fiel auf die Klasse Kundengruppe, die eine Menge von Kunden unter einer bestimmten Kategorie zusammenfasst (siehe Abbildung 28). Auf detaillierte Informationen und Zusammenhänge, die sich hinter einer konkreten Person vom Typ Kunde verbergen, wurde somit bewusst verzichtet, um einen geeigneten Abstraktionsgrad des Modells zu gewährleisten. Alle Informationen, die über die Kundengruppe hinausgehen, sind Aufgabe der konkreten Fachbereiche des Unternehmens und spielen keine Rolle für das IT-Management. Die Klasse Kundengruppe reicht dafür aus, um den Einfluss der Kunden auf die Unternehmung auszudrücken. Beispiele für Kundengruppen sind Privatkunden, Firmenkunden oder Immobilien AGs. Jede Kundengruppe wird durch eine kurze Beschreibung sowie den Beziehungen zu Bankprodukten charakterisiert. Ebenfalls können sie durch Geschäftsfelder, die eine bestimmte Hierarchie von Organisationseinheiten darstellen, gekennzeichnet werden.

Das Unternehmen selbst kann als eine Organisationseinheit aufgefasst werden, die aus mehreren untergeordneten Einheiten bestehen kann. Eine Organisationseinheit fasst dabei mehrere Stellen zusammen, wobei es sich bei einer Stelle um die kleinste organisatorisch zu definierende Organisationseinheit handelt (vgl. Abbildung 28).

Eine Stelle (z.B. Systemspezialist für Microsoft Office Produkte) ist weder räumlich noch zeitlich festgelegt und wird in der Regel personenunabhängig definiert, damit sie von der Aufgabe her determiniert und von einem Stellenwechsel einer konkreten Person nicht bedroht ist. Aus diesem Grund wird der Stelle über das so genannte Rollenkonzept eine Person zugeordnet.

In der Rolle (z.B. Architekt) werden alle für die Stelle benötigten Qualifikationen definiert, die von der die Rolle besetzenden Person erfüllt werden sollten. Um die Abweichung zwischen den geforderten und den tatsächlich von der Person erfüllten Qualifikationen unterscheiden zu können, wurde sowohl

die Klasse Rolle als auch die Klasse Person mit der Qualifikation in Beziehung gesetzt, so dass mögliche Schulungsmaßnahmen für die Person ermittelt werden können.

Das Paket der Geschäftsmodell-Ebene weißt ähnliche Objekte und Konstrukte auf, die bereits zahlreich in den Arbeiten von Scheer Verwendung finden (vgl. [Sc01, Sc02]).

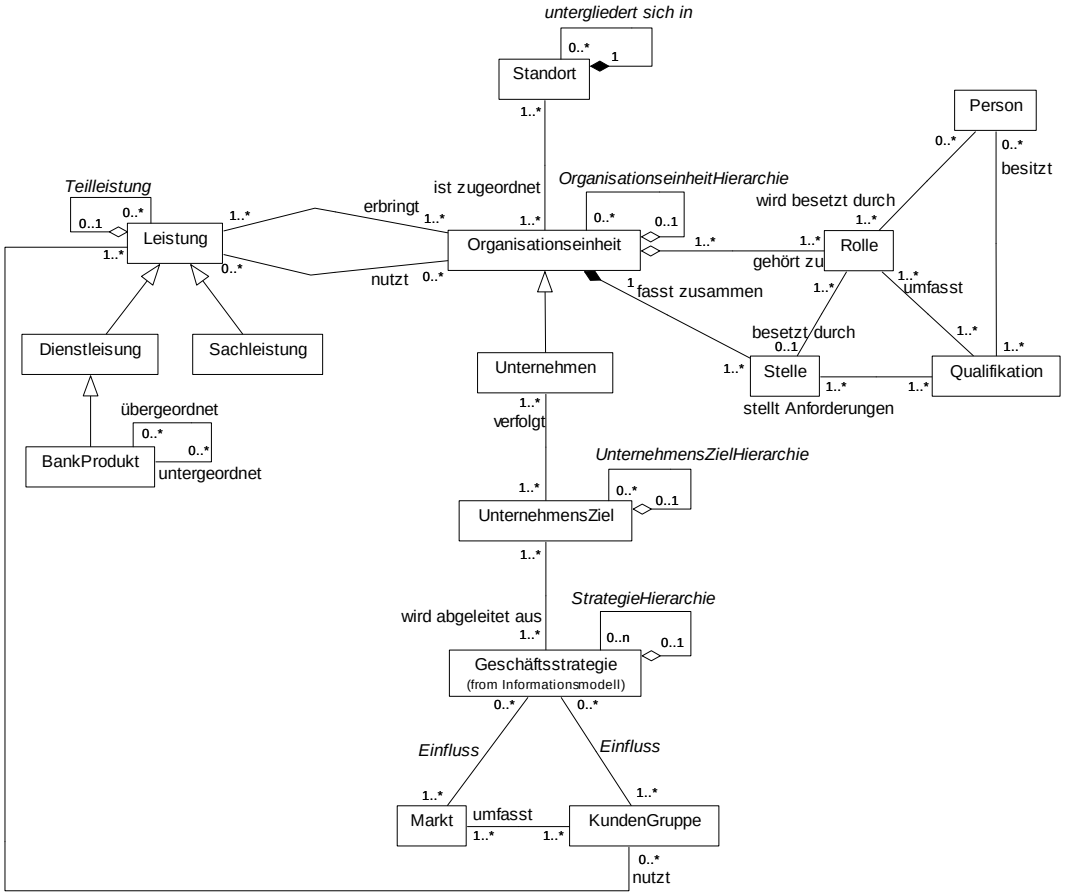


Abbildung 28: Paket der Geschäftmodell-Ebene

5.2.1.2 Paket der Geschäftsprozess-Ebene

Auf der Geschäftsprozess-Ebene werden die Aktivitäten der Geschäftsfelder (z.B. Geschäftsfeld Deutschland) durch ihre Teilprozesse beschrieben, wie sie über Leistungsbeziehungen miteinander verknüpft sind und wie sie in Verbindung mit Organisationseinheiten stehen [HVB03a].

Die entscheidende Klasse dieses Teilmodells ist die des Geschäftsprozesses (z.B. Vertrieb, Wertpapier Abwicklung). Sie kapselt die Logik der zu erbringenden fachlichen Leistung und kann Teil eines Bankproduktes sein (siehe Abbildung 29). Ein Geschäftsprozess ist einer oder mehreren Organisationseinheiten zugeordnet und darf von bestimmten Rollen ausgeführt werden. Darüber hinaus können sie sowohl einer Vorgänger-Nachfolger, als auch einer hierarchischen Ordnung unterliegen. Besitzt ein Geschäftsprozess bzw. dessen Teilprozesse atomaren, transaktionsgesicherten und widerverwendbaren Charakter, so bezeichnet man sie als Geschäftsfunktion.

Der Übergang von Geschäftsprozess zu Geschäftsfunktion ist jedoch recht schwammig und wurde in einigen Gesprächen diskutiert. Das dennoch eine Einführung dieser beiden Klassen gerechtfertig ist, soll die Definition von Bonsangue et al. [Bo05] untermauern. Nach ihr muss ein Geschäftsprozess ein Resultat für einen Kunden liefern, eine Geschäftsfunktion bietet dahingegen lediglich eine nutzvolle Funktionalität für ein oder mehrere Geschäftsprozesse an.

Die Modellierung des Modells der Geschäftsprozess-Ebene orientierte sich, wie das Geschäftmodell-Paket, an den Arbeiten von Scheer (vgl. [Sc01, Sc02]). Zudem leistet es einen wesentlichen Beitrag zum Paradigma der Serviceorientierung, da bei der Modellierung von Geschäftsprozessen und -funktionen bewusst auf eine Aggregations- bzw. Kompositionsbeziehung verzichtet wurde. Aufgrund der Kardinalität 1..n der Teilprozess Assoziation und der Beziehung zu der Geschäftsfunktion ist es möglich, diese in mehreren übergeordneten Geschäftsprozessen wieder zu verwenden. Bei einer Komposition wäre dieser Sachverhalt nicht abbildbar. Auch wenn die Geschäftsprozesse bislang nur einem übergeordnetem Prozess zugeordnet sind, ist ein deutlicher Trend zur Wiederverwendung sichtbar.

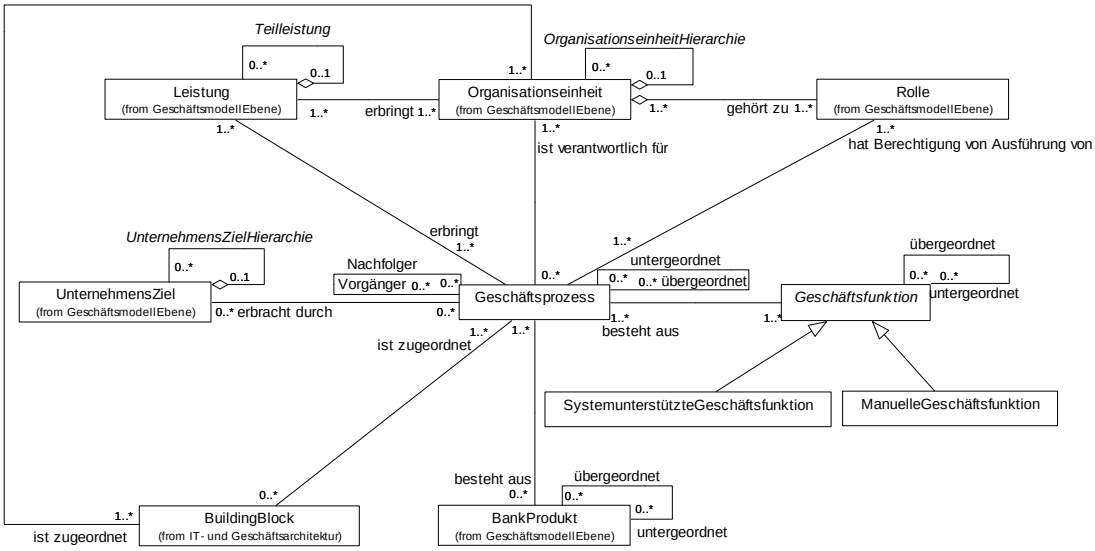


Abbildung 29: Paket der Geschäftsprozess-Ebene

5.2.1.3 Paket der Anwendungssystem-Ebene

Das Teilmodell für die Anwendungssystem-Ebene setzt die Anwendungssysteme sowohl in einen technischen, als auch betriebswirtschaftlichen Kontext [HVB03a]. Über Beziehungen zu beispielsweise den Geschäftsprozessen oder den technischen Produkten, wird eine enge Verknüpfung zu weiteren Ebenen deutlich.

Kern des Modells ist die Anwendungslandschaft mit ihren Anwendungssystemen[26], die jeweils in mehrere Teilsysteme[27] untergegliedert sein können (siehe Abbildung 30). Anwendungssysteme müssen den ihnen gestellten Anforderungen gerecht werden, in dem sie u.a. gewisse Funktionalitäten, d.h. Use-Cases bereitstellen, mit deren systemunterstützte Geschäftsfunktionen abgewickelt werden können. Da *Legacy* Systeme (sehr alte Anwendungssysteme) nicht immer einer Geschäftsfunktion zugeordnet werden können trägt die Kante zwischen Use-Case und systemunterstütze Geschäftsfunktion auf der Seite der Geschäftsfunktion die Kardinalität 0..1. Der Use-Case spezifiziert zudem, wann und wie auf ein Geschäftsobjekt zugegriffen wird.

Der Begriff Anwendungssystem besitzt zunächst rein organisatorischen Charakter, indem Attribute wie z.B. der Name, Bereich oder die Risikoklasse allgemeine Auskunft über ein Anwendungssystem geben. Anwendungssysteme einer Bank unterliegen zudem Anforderungen der Bundesanstalt für Finanzdienstleistungsaufsicht, dessen Hauptziel darin besteht, Missständen im Kreditwesen entgegenzuwirken. Sie gibt Banken Regeln vor, die bei der Gründung und dem Betreiben der Geschäfte zu

[26] Z.B. ein Anwendungssystem zur Erfassung von Überweisungsaufträgen

[27] Das Anwendungssystem „Depotverwaltung“ besitzt beispielsweise ein Teilsystem „Depoteröffnung“.

beachten sind [Baf05]. Informationen über bzw. Eigenschaften eines Anwendungssystems die relevant für die Bankenaufsicht sind, werden im Modell zunächst unter dem Attribut Bankenaufsichtsinformationen zusammengefasst. Zukünftig sollen anstelle dessen konkrete Attribute eingeführt werden, die beispielsweise die erforderliche Berücksichtigung konkreter Gesetzte (z.B. MAK, d.h. Mindestanforderungen an das Kreditgeschäft) angeben.

Informationen, welche technischen Produkte, welche Frameworks, Plattformen oder Umgebungen im Zusammenhang mit einem Teilsystem oder Anwendungssystem stehen, können implizit über die Verknüpfung zu den Technologiesets und somit über Assoziationen der System-Ebene (siehe 5.2.1.5) hergeleitet werden. Welche Produkte und welche Hardware für den Einsatz eines Anwendungs- bzw. Teilsystems erforderlich sind, kann über die Betriebs-Ebene (siehe 5.2.1.6) ermittelt werden.

Wie bereits oben angerissen, arbeiten Anwendungssysteme eines Unternehmens nicht autark sondern sind in einem komplexen Netz miteinander verwoben. Benötigt ein Anwendungssystem zur Erbringung seiner fachlichen Funktion die Unterstützung eines anderen Anwendungssystems, so gehen sie eine Verbindung ein, in der ein Anwendungssystem die Rolle des Nutzers und das andere die des Anbieters einnimmt (vgl. Abbildung 30).

Sollen Verbindungen feingranular beschrieben werden, so bedarf es einer detaillierten Erfassung der beteiligten Objekte. Zu ihnen zählen die fachlichen und die technischen Konnektoren, die die Stellen von Teilsystemen bzw. Softwarekomponenten beschreiben, über die die Verbindung von dem anbietenden und nutzenden Teilsystem abgewickelt wird.

Fachliche Konnektoren liefern alle Angaben aus fachlicher Sicht. In ihnen wird definiert, welche Verfügbarkeit bzw. Bandbreite für die Verbindung gefordert wird, welche Vor- und Nachbedingungen bei der Funktionsnutzung eingehalten werden müssen und ob ein lesender oder schreibender Zugriff stattfindet. Ein Anwendungssystem kann dabei den gleichen Use-Case, der durch einen fachlichen Export-Konnektor angeboten wird, mittels verschiedener Technologien exportieren. Für einen fachlichen Export-Konnektor können somit mehrere technische Export-Konnektoren existieren. Ein technischer Konnektor entspricht der technisch konkret existierenden Schnittstelle.

Sowohl fachliche als auch technische Konnektoren lassen sich hinsichtlich Export-Konnektoren des Funktion Anbieters und Import-Konnektoren des Funktion Nutzers unterscheiden, bei denen die oben angesprochenen Attribute entsprechend zugeordnet werden (siehe Abbildung 30). Die Export-Konnektoren sind immer dem Use-Case anbietenden Teilsystem zugehörig, z.B. bei einem lesenden Zugriff dem Teilsystem, dass die Daten an den Client liefert oder bei einem schreibenden Zugriff, dem Teilsystem, das die Daten weiterverarbeitet. Über die fachlichen Export-Konnektoren wird dabei definiert, welche Use-Cases des anbietenden Systems über die Verbindung genutzt werden können. Import-Konnektoren sind immer dem nutzenden Teilsystem zugeordnet.

Damit der technische Import-Konnektor des Nutzers mit dem technischen Export-Konnetkorts des Anbieters verbunden wird, bedarf es unter Umständen zusätzlicher Anwendungssysteme, die als Middleware dienen.

Die Hierarchische Beziehung von Anwendung- und Teilsystemen wird durch die Softwarekomponenten[28] des Teilsystems fortgesetzt, die wiederum aus ein oder mehreren Software-Elementen[29] bestehen kann (siehe Abbildung 30). Die beiden zuletzt genannten Klassen bieten den Ansatzpunkt für eine detaillierte Beschreibung der tatsächlich existierenden Software mit ihren Bestandteilen. Begriffe wie Enität, Datenbank-Objekte, Satzdefinition, Datenbank-Tabelle, Klasse mit Methoden und Attributen und Items spielen für eine vollständige Beschreibung von Standard- und individuellen Bankprodukten eine entscheidende Rolle und sollen zukünftig im Repository erfasst werden.

Anwendungssysteme, Teilsysteme und Komponenten besitzen einen Release-Status (mit anderen Worten eine Version), in dem angegeben wird, ob sich das Objekt in Planung, Entwicklung, Test oder im Einsatz befindet bzw. sogar bereits ausrangiert und somit archiviert wurde. Bei der Entwicklung des Modells stand zur Diskussion, eigene Entitäten für die Objekt Versionen anzulegen. Da auf der gewählten Abstraktionsebene jedoch allein das Attribut Status von Interesse ist und es zudem beson-

[28] Z.B. im J2EE Umfeld ein Modul.

[29] Z.B. im J2EE Umfeld ein Item.

ders schwierig ist, klar abtrennbare Definitionen für Objekt-Versionen zu definieren, wurde auf eigenständige Entitäten verzichtet und der Sachverhalt in Form eines Attributes abgebildet.

Die Herausforderung in der Erstellung dieses Teilmodells lag in der Schnittstellenthematik, die die Beziehungen zwischen Anwendungssystemen beinhaltet. Das Teilmodell basiert auf Konzepten, die von Matthes et al. [MW04b] entwickelt wurden und entstand iterativ nach zahlreichen Diskussionen und Abstimmungen innerhalb der HVB Systems. Das vorliegende Modell erwies sich zum Ende dieser Arbeit als stabil und wird unterschiedlichen Interessen von fachlichen und technischen Stakeholdern gerecht. In Abschnitt 6.1 wird die Thematik erneut aufgegriffen und Konzepte für eine graphische Repräsentation dieses Paketes erarbeitet.

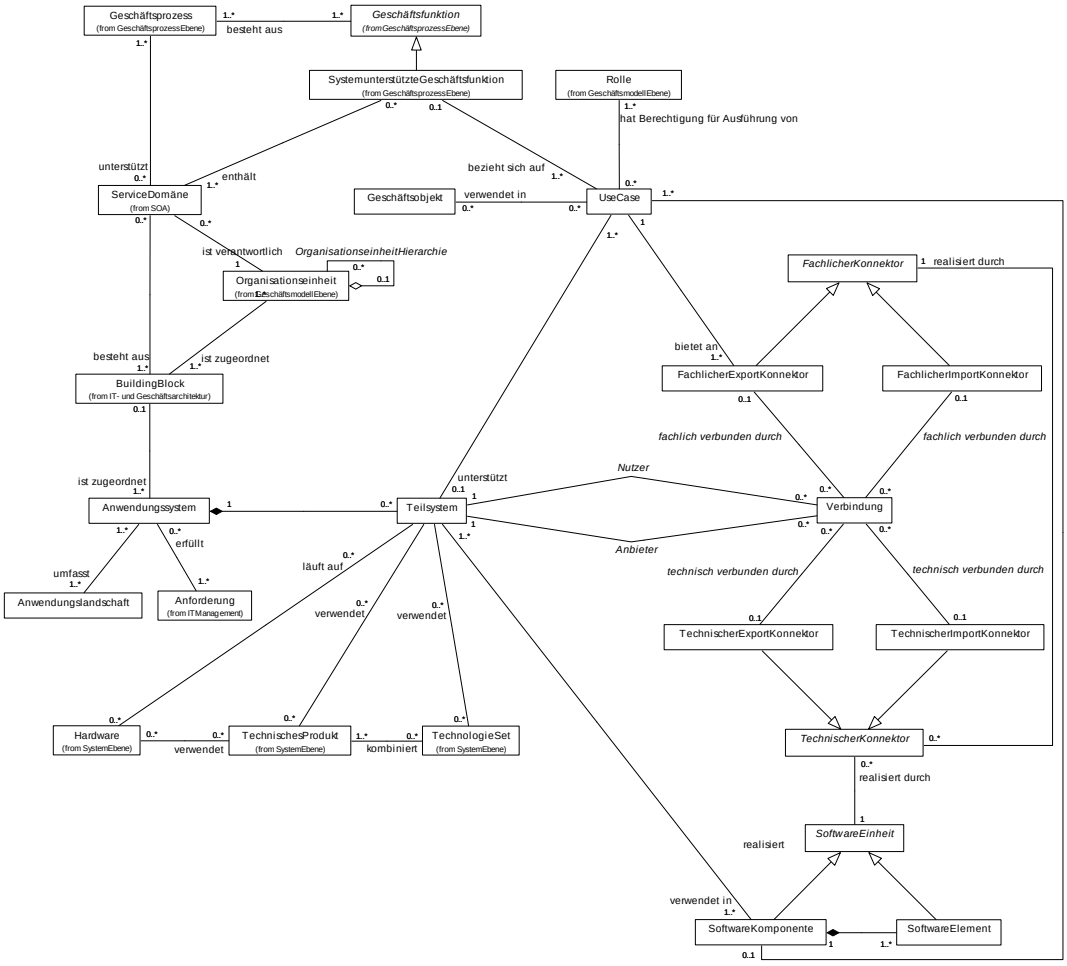


Abbildung 30: Paket der Anwendungssystem-Ebene

5.2.1.4 Paket der Integrations-Ebene

Die Integrations-Ebene befindet sich unterhalb der Anwendungssystem-Ebene, und beschreibt einerseits, welche technischen Elemente notwendig sind, um Anwendungssystem-Komponenten auf Basis der fachlichen Modelle und Prozesse miteinander zu verbinden [HVB03a]. Andererseits schlägt sie die Brücke zwischen der Anwendungssystem und der System-Ebene.

Auf der Integrations-Ebene sind zwei Referenzarchitekturen von besonderer Bedeutung, die *Enterprise Application Integration* (EAI) und die serviceorientierte Architektur (SOA). Es handelt sich um zwei sich ergänzende Architekturansätze, die sich auf verschiedene Ebenen des 6 Ebenen-Modells beziehen (siehe Abbildung 31).

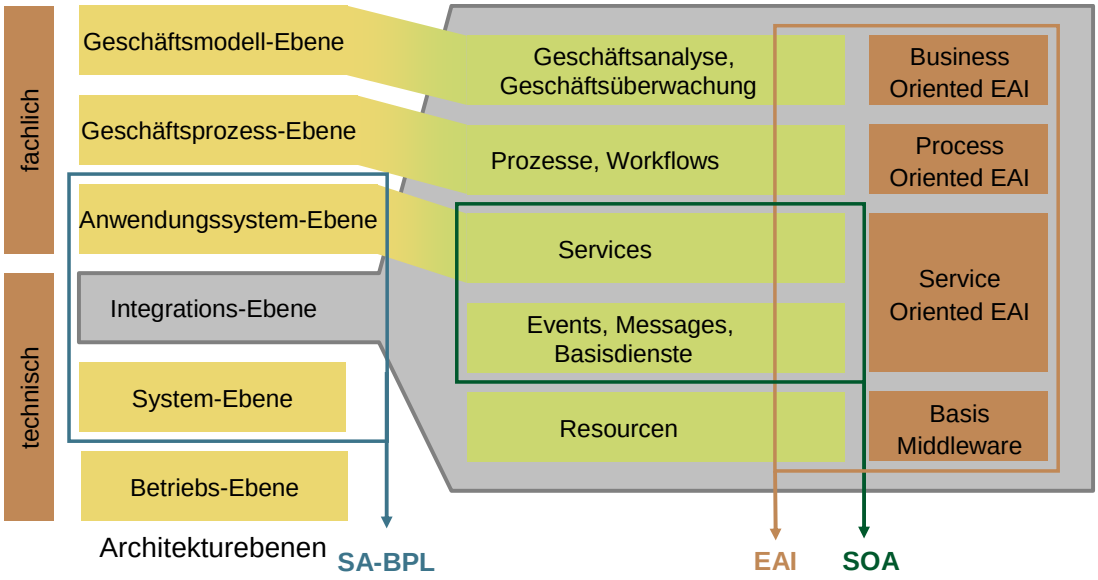


Abbildung 31: EAI und SOA veranschaulicht am 6 Ebenen-Modell angelehnt an [HVB05d]

Wie die obige Graphik zeigt, ist mit dem Begriff EAI eine Integrationsfragestellung verbunden, die sich über mehrere Stufen der Referenzarchitektur erstreckt und auf der Integrations-Ebene Dienste zur Integration der fachlichen Schichten des 6 Ebenen-Modells zur Verfügung stellt [HVB05d]. Der Begriff SOA dagegen bedeutet eine Softwarearchitektur, die auf der EAI-Plattform der Stufe 2 (Service Oriented EAI siehe 5.2.1.4.1) aufsetzt und mittels welche die Anwendungssysteme auf der Anwendungssystem-Ebene die eigentlichen fachlichen Funktionalitäten zur Verfügung stellen [HVB05d].

Lässt man die eher technischen Aspekte für die Integration von Anwendungen sowie die Ebenen übergreifende Bereitstellung von Funktionen außen vor und greift den Begriff der Integration als einen die Anwendungssystem Architektur betreffenden Aspekt auf, stellt sich die Frage nach dem Zusammenhang von Anwendungssystemen und den in ihnen bzw. den bei dessen Herstellung verwendeten Produkten. Der Softwarearchitektur-Bebauungsplan (siehe Anhang A.2) soll diese Frage auf der Integrations-Ebene beantworten, indem er Softwarearchitekturen über ein Schichtenprinzip und Klassifikation technischer Produkte beschreibt.

Die Integrations-Ebene definiert somit logische Modelle, hier die der Referenz- und Softwarearchitekturen, für die physikalischen Elemente der Architektur, d.h. den Technologien selbst (vgl. [SC04a]).

5.2.1.4.1 EAI

Unter dem Schlagwort *Enterpise Application Integration* (EAI) versteht man laut Buhl et al. [BP01] die unternehmensweite Integration von Software-Anwendungen. Mittels EAI soll eine gemeinsame Ebene für unterschiedliche Programme/Anwendungen und eine gemeinsame Basis für verteilte Anwendungen in Unternehmen geschaffen werden. "EAI stellt dabei die Kommunikationsschnittstelle zwischen den unterschiedlichen Technologien, Standards, Datenformen und Protokollen dar. Im Unterschied zur reinen Datenintegration setzt EAI eine Ebene höher an und verbindet Logik und Geschäftsprozessanwendungen. Dabei wird auch die Semantik der Daten berücksichtigt. Die Verbindung der Daten erfolgt nämlich nicht nur auf thematischer Ebene, sondern auch auf betriebswirtschaftlicher, indem verschiedene Geschäftsprozesse miteinander verknüpft werden."[BP01]. Nach Bernhard et al. [BBB03] kann die Integrationsfähigkeit als ein wichtiger Maßstab zur Beurteilung der Leistugnsfähigkeit der IT-Infrastruktur dienen.

Ackermann et al. [HVB05d] greifen den Begriff EAI als einen methodischen Ansatz zur strukturierten, systematischen und strategischen Herangehensweise an die Integration von Anwendungen auf. Er unterscheidet sich von konventioneller Integrationsmethodik dadurch, dass Integration in einem projektübergreifenden und strategischen Kontext betrachtet wird. Aufbau von Architekturen, in denen heterogene Anwendungen und Systeme flexibel und kostengünstig interoperieren können, ist Aufgabe der EAI. Um dieses Konzept für die Integration von Anwendungen im Unternehmen umzusetzen, eig-

net sich laut Ackermann et al. [HVB05d] ein Satz verschiedener Technologien, die wie bereits in Abbildung 31 dargestellt, in vier verschiedene Integrationsstufen eingeteilt werden (vgl. Abbildung 32):

1. *Middleware* sind die grundlegenden Integrationstechnologien, welche die Basis für die anderen Stufen bzw. für die konventionelle Integration bilden.
2. *Service Oriented EAI* Technologien ermöglichen dynamische Integration über einen zentralen Kommunikationsbus mit Funktionen wie dynamisches Routing, Nachrichtentransformation und Ereignissteuerung.
3. *Process Oriented EAI* Technologie erlaubt die Steuerung und Kontrolle von Geschäftsprozessen über Anwendungsgrenzen hinweg.
4. *Business Oriented EAI* Technologien ergänzen prozessorientierte EAI um Prozessmonitoring und Business Intelligence Funktionalitäten.

Der Ansatz von Ackermann et al. [HVB05d] bildet die Grundlage für die EAI Komponenten des Integrations-Paketes, dessen grundlegende Philosophie das Baukasten-Prinzip ist. Statt einer großen, unspezialisierten Integrationsarchitektur, die alle möglichen und projektspezifischen Integrationsszenarien beschreibt, werden kleine, spezialisierte Integrationspatterns, und -bausteine für elementare Szenarien und Aufgaben bereitgestellt, die miteinander verknüpft werden können, um komplexere Integrationsszenarien durchzuführen (siehe Abbildung 32). Integrationsbausteine, und somit auch die Referenzarchitektur, werden dabei zunächst produktneutral definiert. In einem aufbauenden Schritt muss mit Hilfe technischer Produkte eine produktspezifische Referenzarchitektur umgesetzt werden.

Beispielhaft kann das Konzept anhand des SharedData Integrationsbausteins erläutert werden. Hierbei handelt es sich um einen Baustein der Stufe 1 „Middleware“, der Funktionsgruppe Kommunikation. Die Funktion dieses Bausteins ist der Austausch von Informationen über gemeinsam genutzte (engl. shared) Datenbereiche, wie beispielsweise Datenbanken, Files oder gemeinsam genutztem Speicher (engl. shared memory). Die Dienste, die von diesem Baustein zur Verfügung gestellt werden sind beispielsweise die Synchronisation von Daten, Anlegen, Löschen oder Lesen von Einträgen. Der SharedData Baustein ist abhängig vom ServiceFassade Integrationsbaustein, der wieder verwendbare Funktionen für beliebige Servicenehmer zur Verfügung stellt. Ebenso besteht eine Abhängigkeit zum TechnicalConfigurationDirectory Integrationsbaustein, der Ressourcen und Funktionen definiert, die von anderen Bausteinen bei der Ausführung ihrer Funktionen verwendet werden. Charakterisierend für den ShardeData Baustein ist, dass er in der Regel mit anderen Kommunikationsmechanismen z.B. Request-Reply kombiniert wird. Attribute geben beispielsweise die Lebensdauer der Daten an, temporär/langlebig, oder die verwendeten Synchronisationsverfahren, z.B. Sperren.

Anwendungssysteme können entweder die Bausteine einer Referenzarchitektur verwenden und durch dessen Implementierung Funktionalitäten bereitstellen, oder Bausteine anderer Anwendungssysteme verwenden, um Funktionalitäten in Anspruch zu nehmen.

5.2.1.4.2 SOA

Laut Natis et al. [NS03] ist die serviceorientierte Architektur (SOA): "…a best-practice architecture pattern for the systematic design of request/reply applications. Its primary intentions are business-level software modularity and rapid, nonintrusive reuse of business software in new runtime contexts. Users must understand the essence of SOA, as well as its strength and limitations, to identify its role in the overall architecture of modern enterprise IT."

Ackermann et al. [HVB05d] ordnen die SOA Referenzarchitektur auf der Integrations-Ebene an, da sie auf EAI aufsetzt und mittels Umsetzung durch die Anwendungssysteme auf der Anwendungssystem-Ebene fachliche Funktionalitäten zur Verfügung stellt (vgl Abbildung 32). SOA besteht laut Popescu et al. [HVB04d] aus grobgranularen Softwareeinheiten, die geschäftlich-fachliche Funktionen anbieten und mit anderen Anwendungen und Services durch ein lose gekoppeltes, nachrichtenbasiertes Modell kommuniziert. In einer SOA sind Anwendungen als eine Menge voneinander unabhängigen Services zu implementieren, die wohldefinierte Schnittstellen zur Verfügung stellen. Demzufolge handelt es sich bei der SOA um eine Softwarearchitektur, bei der Funktionen mit bestimmten Eigenschaften auf eine definierte Art für Servicenehmer zur Verfügung gestellt werden.

Im Bezug auf die EAI Referenzarchitektur wird eine Servicedomäne[30] durch Implementierung mehrerer Bausteine realisiert. Im Laufe der Entwicklung des Teilmodells für die Anwendungssystem-Ebene zeigte sich, dass alle weiteren Komponenten, die für die Umsetzung einer SOA und somit relevant für ein SOA Repository sind, Objekten der Anwendungssystem-Ebene entsprechen und nur SOA spezifische Attribute ergänzt werden mussten.

5.2.1.4.3 Softwarearchitektur

Jedem Anwendungssystem liegt eine Softwarearchitektur zugrunde, die die Struktur des Anwendungssystems durch Systemkomponenten und ihren Beziehungen untereinander beschreibt [Ba00]. Unter einer Systemkomponente versteht Balzert [Ba00] einen abgegrenzten Teil eines Softwaresystems, welcher als Baustein für die physikalische Struktur einer Anwendung dient. Besteht ein Anwendungssystem und somit die zugrunde liegende Softwarearchitektur aus vielen Komponenten, bedarf es eines Strukturierungskonzeptes, wie beispielsweise eine Schichtenarchitektur. Diese beinhaltet klar definierte Schichten, denen Bausteine zugeordnet werden.

Die Softwarearchitektur der HVB Systems Integrations-Ebene ist der Softwarearchitektur-Bebauungsplan, der Softwarekomponenten (Bausteine) vordefinierten Schichten zuordnet (siehe Anhang A.2). Im Allgemeinen sollte ein Baustein genau einer Schicht zugeordnet werden. Die Praxis zeigte jedoch, dass es bei den Schichten des Softwarearchitektur-Bebauungsplans dazu kommen kann, dass Bausteine mehreren Schichten zugeordnet werden müssen (siehe Anhang A.2). Aus diesem Grund ist die Kardinalität zwischen dem Baustein und der Schicht seitens der Schicht 1..n (vgl. Abbildung 32).

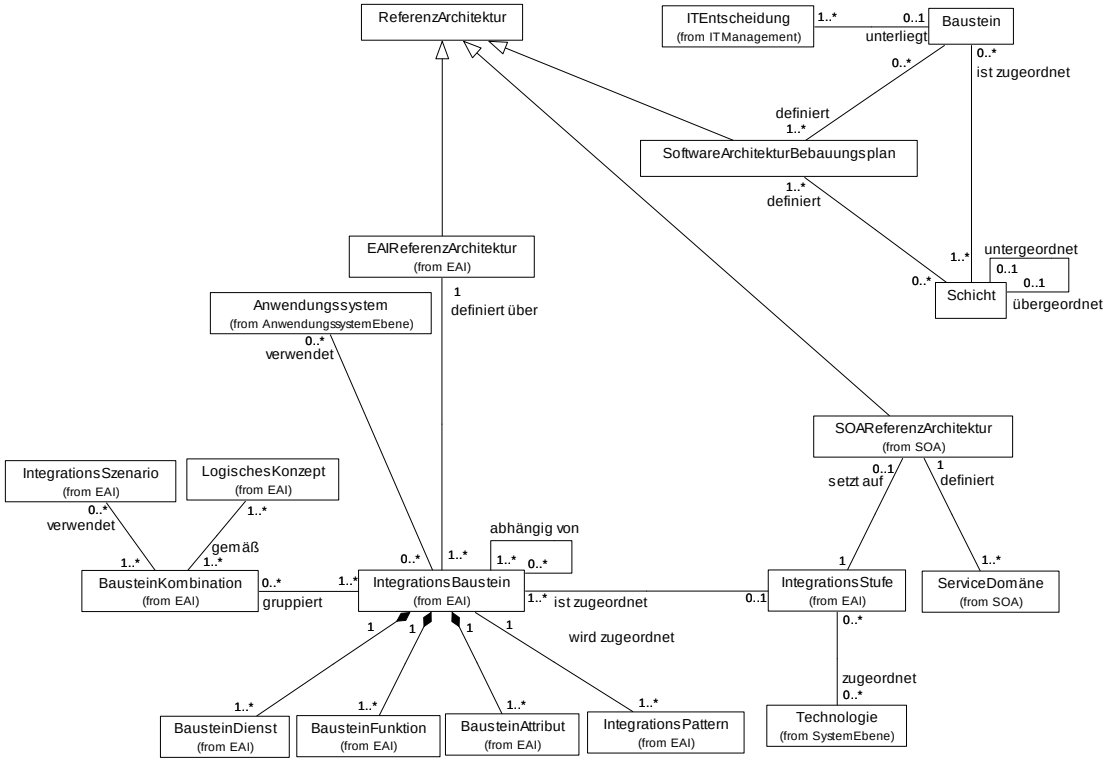


Abbildung 32: Paket der Integrations-Ebene

Jeder Baustein bündelt strategisch Anforderungen und beinhaltet eine Gruppierung von Diensten, die einer übergeordneten Schicht zur Verfügung gestellt werden. Die Schichten sind, bis auf wenige Aus-

[30] Eine Servicedomäne ist eine Verwaltungseinheit, die sich über eine Menge von Services definiert.

nahmen, linear geordnet, d.h. werden entsprechend ihrem Abstraktionsniveau angeordnet und können stets nur auf die nächst Niedrigere zugreifen. Vorraussetzung für eine solche Schichtenarchitektur ist nach Balzert [Ba00] ein natürliches Abstraktionskriterium, nach dem die Komponenten geordnet werden können, und eine Nutzungs-Beziehung der Schicht Dienste. Vorteile einer Schichtenarchitektur sind die Strukturierung in Abstraktionsebenen sowie die Unterstützung von Widerverwendbarkeit, Änderbarkeit und Wartbarkeit. Nachteilig sind der Effizienzverlust, da Daten über verschiedene Schichten transportiert werden müssen und nicht immer eindeutig voneinander abgrenzbare Abstraktionsschichten definiert werden können (siehe Querschnittsbaustein des Softwarearchitektur-Bebauungsplans Anhang A.2).

Es besteht nicht nur eine enge Verbindung zwischen den Referenzarchitekturen SOA und EAI sondern auch zwischen der EAI Referenzarchitektur und dem Softwarearchitektur-Bebauungsplan, da sich die Integrationsbausteine im Softwarearchitektur-Bebauungsplan wieder finden. Als Kritikpunkt lässt sich für die Integrations-Ebene formulieren, dass sie sich im Grunde genau auf die Konzepte der Schichten bzw. Stufen und der Bausteine reduzieren lässt.

5.2.1.5 Paket der System-Ebene

Auf der System-Ebene werden die technischen Komponenten beschrieben, d.h. gültige Technologiestandards und Plattformen, sowie technische Produkte einschließlich ihrer Lebenszyklen [HVB03a].

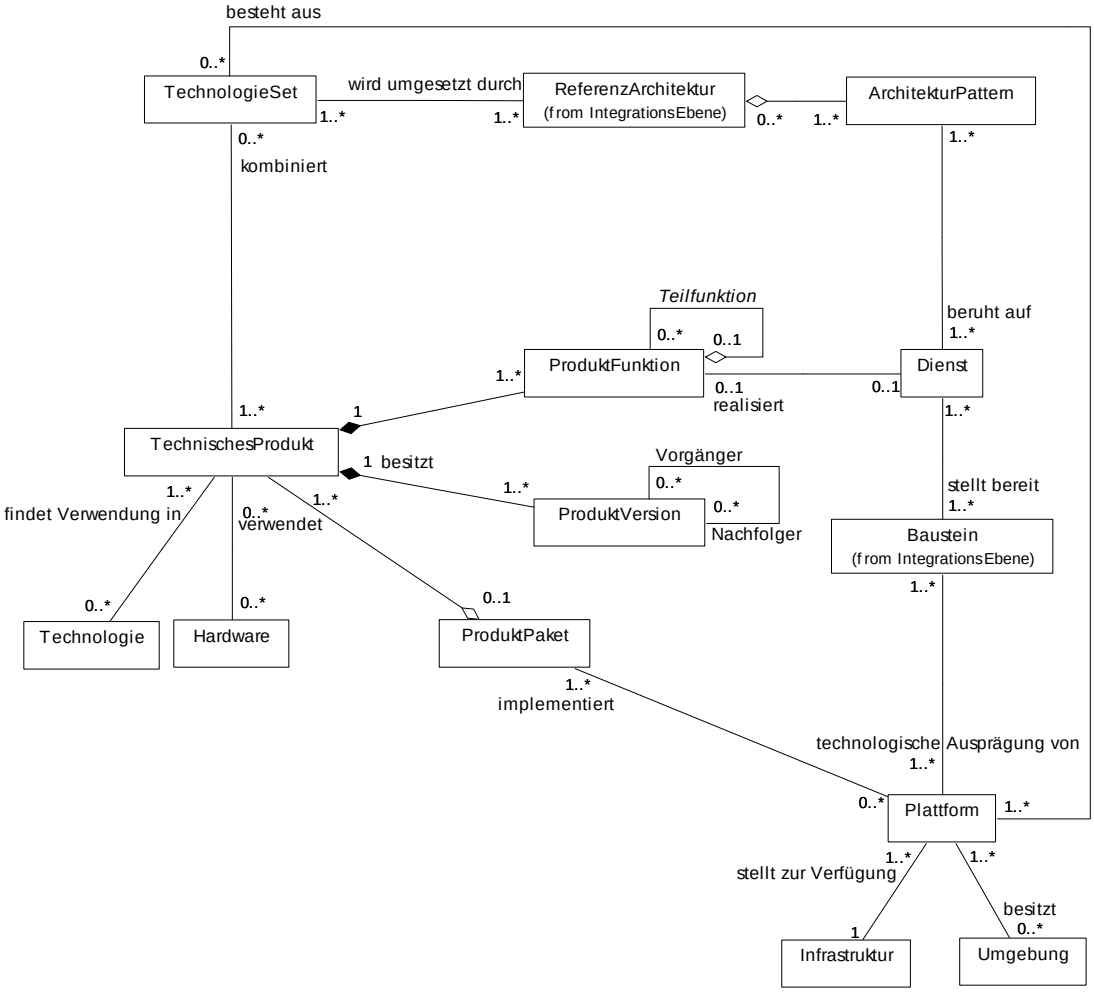


Abbildung 33: Paket der System-Ebene

Dieses Paket dreht sich um die technischen Produkte, die bei der Entwicklung und dem Einsatz von Anwendungssystemen eingesetzt werden und auf einer bestimmten Hardware aufsetzten (siehe Abbildung 33). Wie bereits in der Ausarbeitung der Unternehmensarchitektur eingeführt wurde, bedarf es einer Standardisierung von Produkten, um eine übersichtliche, leicht zu steuernde Landschaft zu erzielen. Für diesen Zweck wurde in der HVB Systems das Konzept der Technologiesets entwickelt (siehe Anhang A.3), die eine gültige Kombination von technischen Produkten inklusive eingesetzter Versionen und Lebenszyklus für die Entwicklung, sowie den Betrieb von Anwendungssystemen beschreibt. Mit Hilfe der Technologiesets ist eine ganzheitliche Sicht auf die in einem bestimmten Anwendungsgebiet eingesetzten Technologien möglich (nähere Ausführung siehe 5.2.2).

Ein Produkt stellt Produktfunktionen bereit, die gewünschte Dienste eines Bausteins[31] erfüllen (vgl. Abbildung 33). Ein Produkt wird auf diese Weise einem Baustein zugeordnet und damit kategorisiert. Produkte, die keine Anforderungen der Bausteine erfüllen, können somit identifiziert und eliminiert werden, so dass eine Bereinigung der IT-Landschaft vorgenommen werden kann.

Welche Betriebselemente für den Einsatz des technischen Produktes erforderlich sind, kann über die Betriebs-Ebene beschrieben werden.

Die System-Ebene entstand auf Basis der Konzepte der HVB Systems unter Berücksichtigung entsprechender Modelle des CIM Modells (siehe *Core und System Specification* in [DMT05]).

5.2.1.6 Paket der Betriebs-Ebene

Auf der Betriebs-Ebene werden Vereinbarungen zwischen unterschiedlichen Parteien bezogen auf unterschiedliche Betriebsmittel festgelegt. Für den Regelbetrieb innerhalb einer Firma sind beispielsweise Verfügbarkeiten, Antwortzeiten und Erfordernisse an Sicherheit und an Backup zu bestimmen [HVB03a].

Die Betriebs-Ebene besitzt im Vergleich zu den anderen Ebenen einen etwas abstrakteren Charakter, da sie Elemente anderer Ebenen mit Hilfe von Oberbegriffen zusammenfasst und neue Beziehung definiert. Aus diesem Grund wurde sie parallel zu den restlichen fünf Ebenen angeordnet (siehe Abbildung 27). Das Teilmodell wurde in Anlehnung an das Context Diagramm der IT-Portfolio Management Facility Specification (ITPMF siehe [OMG04]) entwickelt.

Ein Betriebselement steht stellvertretend für alle betriebsrelevanten Objekte, wie beispielsweise Produkte, Plattformen, Umgebungen oder Hardware (siehe Abbildung 34). Für ein Betriebselement können Vereinbarungen, z.B. über die Betriebszeit, getroffen werden. Dabei können Betriebselemente innerhalb einer Hierarchie gruppiert werden, so dass die Definition von Vereinbarungen und Anforderungen auf hoher Abstraktionsebene, beispielsweise einer Testumgebung inklusive verschiedener Software und Hardwareelementen, und niedrigen Abstraktionsebenen, wie zum Beispiel einem CD Brenner, vorgenommen werden kann.

Bestehen bei einem Betriebselement Abhängigkeiten zu anderen, so können diese mittels der Betriebsmittelhierarchie abgebildet werden. Abhängigkeiten zwischen Anwendungssystemen können dabei mittels Beziehungen der Anwendungssystem-Ebene ermittelt werden, Abhängigkeiten zwischen Produkten mithilfe der Beziehungen auf der System-Ebene.

Eine Vereinbarung umfasst die Pflichten des Anbieters, die Mitwirkungspflichten des Nutzers, das Niveau bzw. die Qualität der benötigten Dienstleistung und gegebenenfalls die Preise je Diensteinheit [BES04]. Man kann bei Vereinbarungen zwischen Standard Vereinbarungen, d.h. solchen die ständig neu abgeschlossen werden, und speziellen Vereinbarungen, d.h. Vereinbarungen, die für ein spezielles Szenario aufgesetzt werden müssen, unterscheiden. Da sich beide Arten durch die gleichen Attribute und Beziehungen beschreiben lassen, reicht für die Modellierung die eine Klasse Vereinbarung. Da sich eine spezielle Vereinbarung an einer Standard Vereinbarung orientieren kann, wurde eine rekursive Beziehung der Klasse Vereinbarung eingeführt. Wird eine Vereinbarung geschlossen, sind mindestens ein Anbieter und ein Nutzer zu definieren (vgl. Abbildung 34).

Ebenso wie die Betriebsmittel, können auch die Beteiligten der Vereinbarung über die Definition von Hierarchien auf hohem bzw. niedrigem Abstraktionslevel beschrieben werden. Das Modell des ITPMF

[31] Damit auch einer Schicht des Softwarearchitektur-Bebauungsplans.

wurde durch die beiden Beziehungen zwischen Vereinbarung und Beteiligter ergänzt, um die Rollen des Anbieters bzw. des Nutzer definieren zu können.

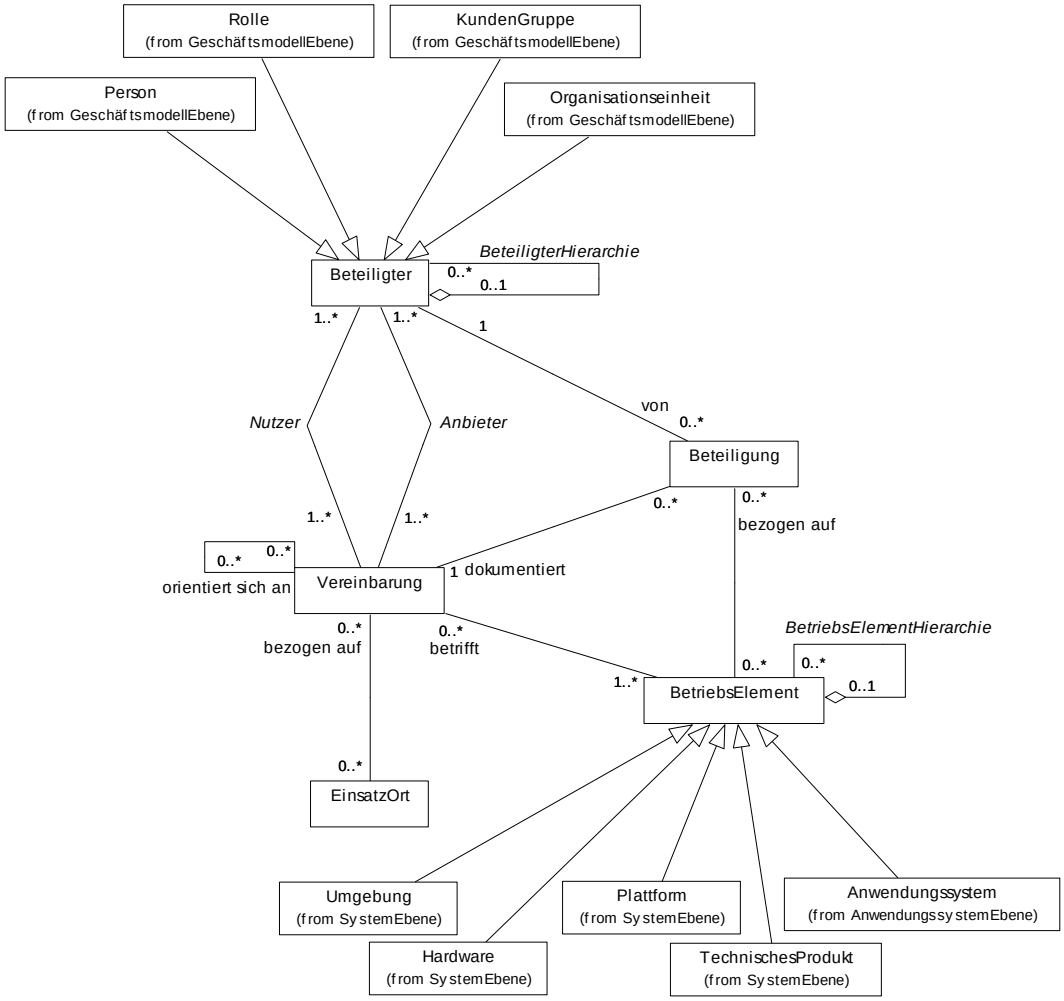


Abbildung 34: Paket der Betriebs-Ebene

5.2.2 IT-Management-Paket

Das IT-Management-Paket setzt auf den Informationsobjekten des IT- und Geschäftsarchitektur-Paketes auf und beinhaltet Objekte, die sowohl das Architekturmanagement betreffen, als auch Aspekte des allgemeinen IT-Managements repräsentieren. Mit Hilfe des Architekturmanagementprozess der HVB Info und HVB Systems, soll im folgenden Abschnitt das IT-Management-Paket eingeführt werden [HVB05c].

Der Architekturmanagementprozess stellt, wie in Abbildung 35 veranschaulicht, einen Regelkreis dar. Er verankert die geordnete Weiterentwicklung der Architektur im Unternehmen, indem über klar definierte Phasen mit entsprechenden Richtlinien, Werkzeugen und Standards die IT Komponenten konzipiert und geplant werden. Fortlaufend muss dabei die Einhaltung der Regeln durch organisatorische Verankerung und geeigneten Verfahren sicherstellt werden.

Nach Gernert et al. [GA02] leitet sich der Begriff des Regelkreis aus der kybernetischen Bedeutung des Wortes Regelung ab. Ausgehend von einem Soll-Wert, wird nach Überprüfung des Ist-Zustandes bewusst auf den Prozess mit dem Ziel der Rückführung auf den Soll-Wert eingewirkt.

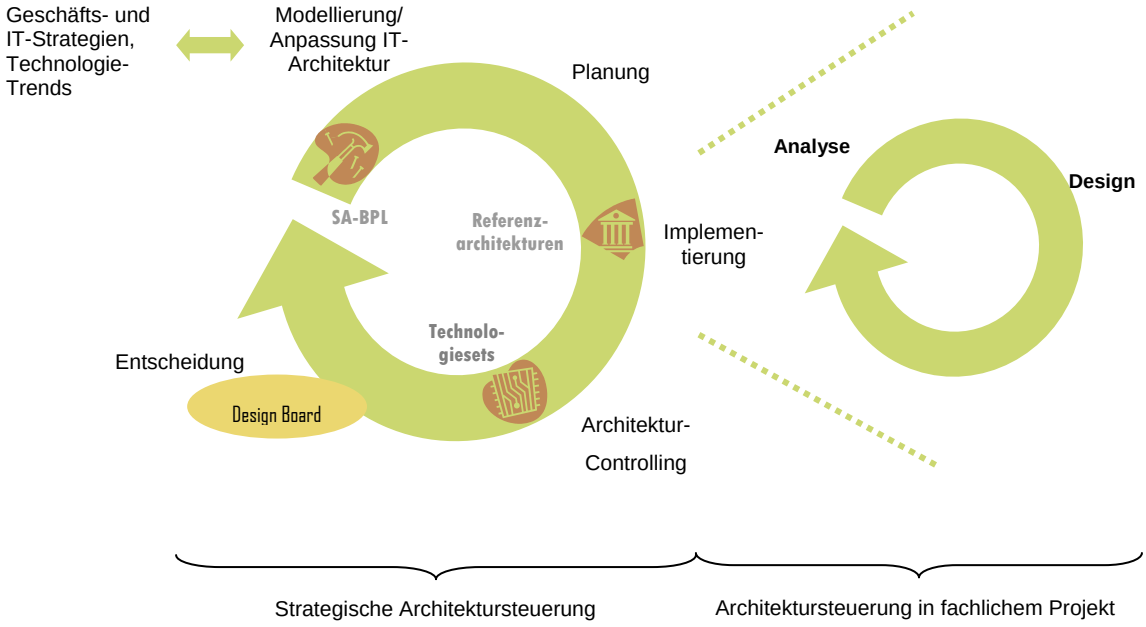


Abbildung 35: Architekturmanagementprozess angelehnt an [HVB05c]

Einen Einstiegspunkt in den Regelkreis bieten die drei externen Einflussgrößen Geschäftsstrategie, IT-Strategie und die Technologietrends, die auf den Architekturmanagementprozess einwirken, aber zugleich auch unter dessen Einfluss stehen, da Ergebnisse und Erkenntnisse die aus dem Architekturmanagementprozess gewonnen werden, auf die IT-Strategie einwirken können (vgl. Abbildung 37). Die Aufgabe des Architekturmanagements besteht darin, die IT-Strategie durch die Bereitstellung von geeigneten Plattformen zu unterstützen und den Return of Investment (ROI) zu erhöhen, indem die Anzahl der Plattformen reduziert sowie einzelne Komponenten wieder verwendet werden, die die Projekte nutzen. Eine konkrete Geschäftsstrategie, wie beispielsweise die end-to-end Optimierung von Geschäftsprozessen, stellt an die IT unter anderem die Anforderung nach der Konnektivität zwischen Teileinheiten [HVB03a].

Die IT-Strategie selbst besteht aus einer Menge von Anforderungen. Zunächst werden auf recht abstrakter Ebene Anforderungen an die Weiterentwicklung der IT-Architektur und der Systemlandschaft gestellt, die von der Bank und deren Geschäftsfelder bezüglich der Umsetzung der Geschäftsstrategie sowie aus technologischen Notwendigkeiten, z.B. Innovationen, unter Berücksichtigung wirtschaftlicher Gesichtspunkte resultieren. Diese Anforderungen werden im Architekturmanagementprozess in konkrete Anforderungen zerlegt, die an Plattformen oder Produkte gestellt werden. Ein Beispiel hierfür wäre die Anforderung „mandantenfähig“, die ein Produkt erfüllen sollte. Anforderungen müssen dabei nicht zwingend von der IT-Strategie stammen, sondern können sich auch aus IT-Projekten ergeben. Wird in einem Projekt eine Plattform eingesetzt, die bislang noch kein Standard ist, aber auch durch keine Standardplattform ersetzt werden kann, so entsteht beispielsweise eine Anforderung nach Standardisierung dieser Plattform. Grundsätzlich sollte dieser Fall jedoch nicht eintreten, da bereits im Vorfeld eine optimale Architektur bereitgestellt werden sollte, die nicht aus den Anforderungen einzelner Projekte entstand, sondern durch vorausschauende Überlegungen und Entscheidungen.

Technologietrends können je nach Größe und Wirkung neben der Geschäfts- und der IT-Strategie auch die Architektur beeinflussen [BES04]. Internettechnologien haben etwa direkten Einfluss auf die Geschäftsstrategie, da mit ihrer Hilfe neue Märkte oder Kanäle erschlossen werden können. Dahingegen wirken neue Technologien, wie die serviceorientierte Architektur (SOA), nur auf die IT-Strategie ein und "kleine" Trends, beispielsweise der Upgrade einer Produktversion, allein auf die Architektur. Der Umfang der Auswirkungen, die dabei ein Technologietrend hervorruft ist jedoch unabhängig seines Wirkungsgebietes. Ein Upgrade, beispielsweise von NT zu XP Systemen, kann einen größeren Aufwand mit sich bringen, als ein Trend, der auf die IT-Strategie einwirkt.

In der Phase *Modellierung/Anpassung* der IT-Architektur werden die Konsequenzen, die aus der Geschäfts- und IT-Strategie für die Architektur gezogen worden sind, über Architekturstandards umgesetzt (vgl. Abbildung 37). Standards werden dabei so entwickelt, dass sie beim Einsatz in IT-Projekten die Umsetzung der strategischen Ziele fördern. Nicht mehr gültige Standards können annulliert, bestehende den neuen Anforderungen angepasst oder von Grund auf neu entwickelt werden. Ein Werkzeug zur Standardisierung von Produkten sich beispielsweise die Technologiesets, die in Projekten eingesetzt werden und bei der Umsetzung der IT-Strategie dienen können. Im folgenden Abschnitt soll dieses Konzept näher erläutert werden.

Eine strategische Prämisse lautet, dass sich die Wettbewerbsfähigkeit von Infrastruktur und Betrieb aus dem Kosten-/ Leistungsverhältnis im Vergleich zum Markt ergibt [HVB03c]. Dabei ist nach Stirmlinger [HVB03c] die Wettbewerbsfähigkeit von Infrastruktur und Betrieb in drei Dimensionen zu gestalten. Erstens die Kosten, da sich angesichts steigenden Kostendrucks Spielräume für marktorientierte Innovationen nur durch Reduzierungen der Betriebskosten erreichen lassen. Dies setzt eine möglichst homogene und technologisch moderne Infrastruktur voraus. Zweitens die Qualität, denn die optimale Unterstützung des Geschäftes, sowie die Minimierung operativer Risiken, setzen eine angemessen leistungsfähige Infrastruktur für Anwendungsentwicklung und Betrieb voraus. Die dritte Dimension stellen die Innovationen dar, da sich angesichts der permanenten technologischen Weiterentwicklung die Leistungsprofile sowie die Kosten-/Nutzenverhältnisse konkreter Technologien permanent verändern. Um nun die Wettbewerbsfähigkeit zu steigern, bedarf es entsprechenden Lösungskonzepten, wie beispielsweise den Technologiesets, die die Effizienz der Architektur stark fördert [HVB03c]. Technologiesets sind eine für die Entwicklung einsetzbare gültige Kombination von Produkten inklusive Versionsnummern und ihren aktuellen Lebenszyklus-Status (siehe Abbildung 36). Sie ermöglichen durch einen Standardisierungsprozess den Fokus auf die am besten geeigneten Plattformen zu lenken. Durch die Festlegung, dass nie mehr als drei Releases eines Produktes parallel produktiv eingesetzt werden dürfen und überalterte Produkte konsequent abgelöst werden, wird die Produktvielfalt deutlich eingeschränkt und eine steuerbare, funktionelle Infrastruktur zu Verfügung gestellt [HVB03c]. Standardisierte, abgestimmte und zielbezogene Technologiesets erleichtern erheblich die Orientierung innerhalb der Entwicklungsprojekte. Existierten bei Einführung des Technologieset-Konzeptes ca. 150 Technologiesets, so konnte durch Konsolidierung auf Versionsbasis und Integration von Lebenszyklen Informationen die Anzahl der Technologiesets deutlich auf 33 reduziert werden. Eine strikte Verwendung der definierten Technologiesets garantiert dadurch eine hohe technische Homogenität und Kontinuität [HVB03c].

J2EE_Server					Letzte Aktualisierung		16.03.2005
					nächste gepl. Aktualisierung		16.09.2005
Technologie-Set			**kritisch**	**Handlungs-bedarf**	**kein Handlungs-bedarf**	**Soll für neue Projekte**	**Ausblick (in Abstimmung)**
Client	Presentation						
	Thin Gui						
		MS Internet Explorer	MS Internet Explorer V5.01 SP1 MS Internet	-	MS Internet Explorer V6.0 SP1	MS Internet Explorer V6.0 SP1	-

Abbildung 36: Ausschnitt eines Technologiesets aus [HVB05c]

Die zweite Phase *Planung* (siehe Abbildung 35) umfasst Projektinitiativen und Design. Aus der ersten Phase Modellierung/Anpassung können Handlungsbedarfe identifiziert werden, die sich aus den Änderungen der Standards ergeben und in Form von neuen (strategischen) IT-Projekten behoben werden. Diese Projekte sind festzulegen und hinsichtlich ihrer Auswirkungen auf die Anwendungslandschaft zu analysieren, da sie häufig Anwendungssysteme verändern oder neue entstehen lassen (vgl. Abbildung 37). Projekte sind mittels eines Namens und einer Projektnummer eindeutig identifizierbar. Mittels eines Hierarchietyps lässt sich dokumentieren, ob es sich um ein Einzelprojekt, Hauptprojekte oder Teilprojekt handelt. Im so genannten Projektkontext wird bei Fachbereisprojekten festgehalten, ob es sich um eine Neu-, Weiter- oder Eigenentwicklung bzw. ein Fremdsoftware-Projekt handelt, bei Plattfomprojekten sind es die Ausprägungen Pilot, Upgrade, Ablösung und Einführung.

Das entscheidende Werkzeug dieser Phase sind die IT-Szenarien, die bereits im Zusammenhang mit dem IT-Bebauungsplan erläutert wurden. Durch die Berücksichtigung möglicher Zukunftssituationen innerhalb der IT einschließlich der Untersuchung relevanter Aspekte der resultierenden Anwendungs-

landschaften, kann die Durchführung eines IT-Projektes festgelegt oder verworfen und anschließend geplant werden.

Implementierung und Projektunterstützung als die dritte Phase des Architekturmanagementprozesses (siehe Abbildung 35), umfasst die konkrete Umsetzung der identifizierten und geplanten IT-Projekte, unter zu Hilfenahme der zur Verfügung gestellten Werkzeuge wie z.B. Referenzarchitekturen oder Technologiesets. Hinter dieser Phase versteckt sich ein zweiter Regelkreis, einem Vorgehensmodell zur Software Entwicklung mit den Phasen Analyse, Design, Implementierung und Entwicklung. Das firmeninterne ALADIN (*Application Development Information Navigator*) ist beispielsweise ein einheitliches, verbindliches Vorgehensmodell für Projekte in der HVB Info und der HVB Systems. Es dient als Leitfaden für Projekte, definiert die einzelnen Aktivitäten sowie die zu erstellenden Ergebnisse und notwendigen Qualitätssicherungsinstrumente und unterstützt iterative Teilprozesse.

Der Implementierung folgt das *Architekturcontrolling* (siehe Abbildung 35). Im Gegensatz zum IT-Projektreview in der vorherigen Phase liegt hier der Fokus auf Architekturstudien und Architekturcontrolling, in denen nicht der konkrete Einsatz der Architekturstandards, sondern die Eigenschaften des Standards an sich und die Gesamtheit aller zur Verfügung gestellten Werkzeuge auf Zweckmäßigkeit, Vollständigkeit und Korrektheit untersucht werden (vgl. Abbildung 37). Für das Architekturmanagement ist es zudem von großer Bedeutung, statistische Auswertungen über die in IT-Projekten eingesetzten Standards und Werkzeuge durchzuführen. Auswertungen beispielsweise auf die Fragen: „In welchem IT-Projekt wurde/wird Plattform X eingesetzt?“; „Wie viele Anwendungen laufen auf der Plattform Y?“ oder „Welche Anwendung läuft auf der Plattform Z?“, dienen dazu, Standards und Werkzeuge hinsichtlich ihres Einsatzes zu bewerten und gegebenenfalls Handlungsbedarfe zu identifizieren. Findet etwa eine Plattform nur in wenigen Projekten Einsatz, obwohl im Vorfeld von einer großen Anzahl an Projekten ausgegangen wurde, ist die Plattform hinsichtlich ihres Aufbaus und ihres Zwecks zu überarbeiten.

Den Regelkreis schließt die Phase der *Entscheidung* (siehe Abbildung 35), wobei sich IT-Entscheidungen auf Standards oder auf abweichende Projektarchitekturen beziehen können. Müssen Standards aufgrund der IT-Strategie oder aus Ergebnissen von Studien geändert werden, bzw. sollen IT-Projekte trotz Abweichung zu den definierten Standards durchgeführt werden, so ist dies in Entscheidungsgremien vorzustellen, abzustimmen und in einem Entscheidungsrepository zu dokumentieren bzw. archivieren (vgl. Abbildung 37). Das *Design Board* ist beispielsweise das Entscheidungsgremium für Architekturentscheidungen der HVB Info und HVB Systems. Die Aufgaben sind die Freigabe des IT-Designs von Projekten (bei Abweichungen vom Softwarearchitektur-Bebauungsplan und/oder Technologiesets), Aufnahme und Abkündigung von Bebauungsplan-Standards sowie Entscheidungen zum Einsatz von EAI in Projekten. Mitglieder des *Design Boards* sind jeweils Geschäftsführer der HVB Systems und der HVB Info, Leiter der Architekturabteilungen und betroffene Building Block Leiter bei Entscheidungen zu Projekten.

Anwendungssysteme bzw. Teilsysteme können sowohl Kosten als auch Vorfälle (engl. Incidents) initiieren (siehe Abbildung 37). Die Aufschlüsselung der IT-Kosten auf die verschiedenen Anwendungssysteme ist ein wichtiger Bestandteil des Architekturmanagements. Durch die Transparenz der verursachten Kosten lässt sich die Auswirkung von Veränderungen bereits in der Planungsphase analysieren, sowie besser und vollständiger beurteilen. Quantitative und qualitative Informationen über angefallene Vorfälle[32], können dabei von entscheidender Bedeutung sein und werden vom IT-Management aus einem eigenständigen System hinzugezogen. Um ein ganzheitliches Bild von der IT zu bekommen, sollte die Betrachtung der Konstenseite jedoch um die der Leistungsseite ergänzt werden[33] [BES04].

Das Teilmodell des Paketes basiert auf dem Architekturmanagementprozess der HVB Systems und Anforderungen die sich aus zahlreichen internen Gesprächen ergaben. Das Modell kann beispielsweise für die Implementierung eines Architekturmanagement Repository verwendet werden, indem alle Informationen über obige Objekte und deren Beziehungen bereitgestellt werden. Zudem fördert das

[32] Z.B. je Anwendungssystem je Kundengruppe.

[33] Z.B. das Aufwiegen der Kosten in Bezug auf die Funktionalität, d.h. den Nutzen eines Anwendungssystems.

Modell die Transparenz und Nachvollziehbarkeit der Prozesse und Abläufe innerhalb der IT. Die Verbindung von Entscheidungen und den beeinflussten Artefakten stellt dabei einen wesentlichen Mehrwert dar.

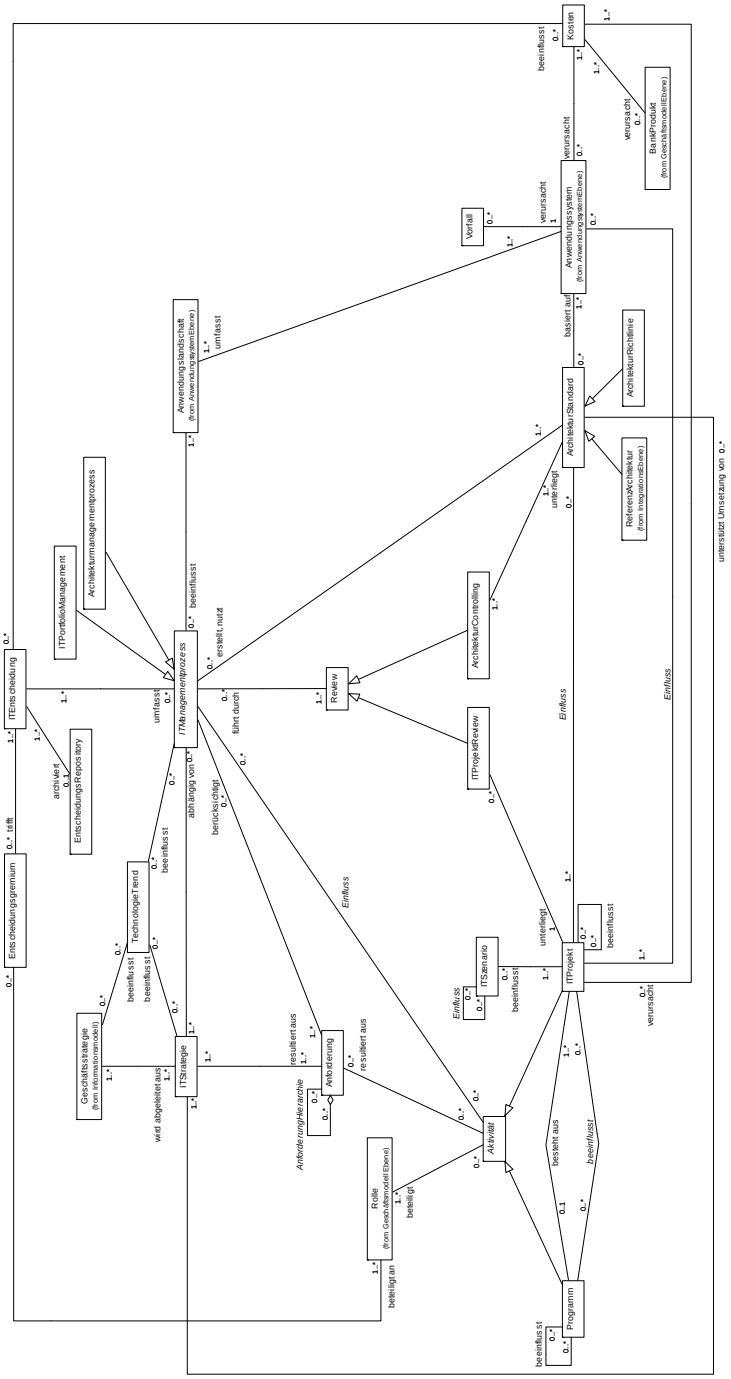


Abbildung 37: IT-Management-Paket

5.2.3 Datentyp-Paket

Für einige Informationsobjektattribute des Modells werden neben den primitiven Datentypen auch Informationsobjekte selbst benötigt. Einleitend wurde bereits darauf hingewiesen, dass einige Attribute aufgrund der Modellübersichtlichkeit und Lesbarkeit stellvertretend für eine Assoziation stehen. Bei diesen Attributtypen handelt es sich jeweils um Objekte, die bereits in den Teilpaketen eingeführt wurden. Eine Organisationseinheit besitzt beispielsweise das Attribut Leiter vom Typ Rolle.

Neben diesem Spezialfall gibt es jedoch auch einige Attribute, deren Typ sich durch eine Klasse spezifizieren lässt, die nicht Teil anderer Pakete ist. Diese Typen werden vom Datentyp-Paket zusammengefasst. Ein Teilsystem besitzt beispielsweise eine Dokumentation, die sich u.a. durch Autor, Version und Ablageort näher beschreiben lässt. Aus diesem Grund wurde die Klasse *Dokumentiertes Wissen* mit den genannten Attributen eingeführt, die immer dann als Attributtypen verwendet wird, wenn es sich um dokumentierte Informationen handelt.

Die beiden Klassen Wissenskategorie und Dokumentiertes Wissen sowie Attribute von diesen Datentypen liefern Hinweise auf die Informationskomponenten des Unternehmens, die beispielsweise in Arbeiten von Scheer [Sc02] und Bonsangue et al. [Bo05] als eigene Domänen modelliert werden.

Abbildung 38 veranschaulicht den Inhalt des Datentyp-Paketes mit den gegenseitigen Beziehungen der Klassen.

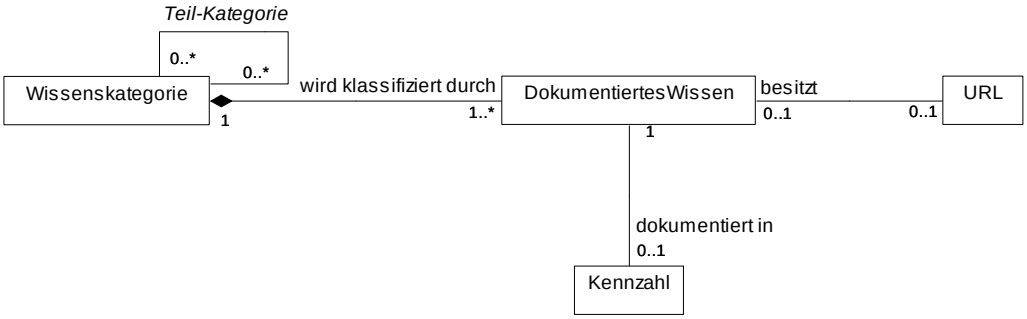


Abbildung 38: Datentyp-Paket

5.3 Kritische Bewertung des Informationsmodells

In diesem Abschnitt soll das erstellte Informationsmodell bezüglich seiner Qualität untersucht werden. Den Rahmen für diese Bewertung stellen die Grundsätze ordnungsmäßiger Modellierung (GoM), die von J. Becker et al. erarbeitet und bereits in Abschnitt 3.2.3.2 eingeführt wurden. Da das Informationsmodell sowohl Charakter eines abbildungsorientierten als auch eines konstruktionsorientierten Modells besitzt, geht die Bewertung auf alle eingeführten Grundsätze ein.

Richtigkeit

Der erste zu hinterleuchtende Punkt ist der Grundsatz der Richtigkeit. Nach Becker et al. [Be00] wird eine semantische Richtigkeit durch die Einhaltung von Namenskonventionen erzielt. Im Informationsmodell wurde bei der Wahl von Objektnamen stets darauf geachtet, eindeutige Benennung der zu modellierenden Objekte zu gewährleisten, um Sprachdefekte wie Synonyme oder Homonyme zu vermeiden bzw. für Äquipollenzen zu sensibilisieren:

- *Synonym*: Unterschiedliche Begriffe, die aber dasselbe Objekt benennen sollen, z.B. Anwendungssystem und Applikation.
- *Homonyme*: Bezeichnungen, die für unterschiedliche Objekte stehen können, z.B. eine Funktion kann ein atomarer Geschäftsprozess oder eine Funktionalität eines Anwendungssystems sein.
- *Äquipollenzen*: Bezeichnungen bzw. Begriffe, die dieselbe Extension, aber verschiedene Intensionen besitzen, d.h. Begriffe die in verschiedener Form denselben Sachverhalt widerspiegeln, z.B. „gleichseitiges Dreieck“ und „gleichwinkliges Dreieck“.

Besonders bei der Bearbeitung der Teilmodelle einzelner Pakete und der Aufnahmen von Anforderungen in Gesprächen mit Mitarbeitern unterschiedlich fachlicher Herkunft, musste besonders auf die Konsolidierung der Begriffe geachtet werden (vgl. Konsolidierung und Sichtenintegration in [KE01]). Eine geeignete Wahl von Objektnamen war jedoch insbesondere für die Geschäftsprozesse, Geschäftsfunktionen, Use-Cases und die Konnektoren schwierig, da keine Klassennamen definiert werden konnten, die ideal für alle Modelladressaten war. Um dennoch für ein einheitliches Verständnis zu sorgen, sammelt das in Anhang C aufgeführte Glossar alle Klassen einschließlich ihrer Bedeutungen und Begriffsdefinitionen.

Neben den Namenskonventionen sorgt die Angabe der Kardinalitäten in der (min, max)-Notation, im Vergleich zur reinen max-Notation, für eine genaue Darstellung des Objektsystems.

Der syntaktischen Richtigkeit wurde das Informationsmodell überwiegend durch das Modellierungswerkzeug Rational Rose (IBM) gerecht, dass bei der Erstellung des Modells nur solche Konstrukte ermöglicht, die den in Abschnitt 3.2.2 eingeführten Konzepten der MOF und UML gerecht werden.

Relevanz

Der Grundsatz der Relevanz lässt sich aus der Perspektive des Modellerstellers schwer beurteilen. Im Verlauf der Modellierung wurden einzelne Teilmodelle mittels beispielhafter Datenerfassung auf die Vollständigkeit des Modellsystems untersucht. Relevante Bestandteile des Objektsystems wurden anhand der Anforderungen aus Interviews der Modelladressaten ermittelt.

An dieser Stelle sei darauf hingewiesen, dass das entwickelte Informationsmodell einen starken Bezug zu Unternehmen des Finanzdienstleistungssektors besitzt. Soll das Modell auf andere Unternehmen angewandt werden, können einige Objekte von geringerer Bedeutung bzw. nicht von Relevanz sein. So besitzt beispielsweise die Klasse Bankprodukt für ein Unternehmen der Automobilindustrie keine Bedeutung. Eine Überarbeitung des Modells aus Sicht der Branche des Unternehmens, sollte für eine „Bereinigung“ des Modells sorgen.

Wirtschaftlichkeit

Es wurde bereits festgestellt, dass sich die Wirtschaftlichkeit eines Modells nur schwer bewerten lässt. Erste Aussagen über den Nutzen des entwickelten Informationsmodells ließen sich jedoch bereits bei dessen Erzeugung feststellen, da die Modellierung zum Verständnis der komplexen Sachverhalte beitrug. Bei Interviewgesprächspartnern zeigte sich, dass die abstrahierte Sicht auf ihre Themengebiete zu neuen Anregungen führen kann. Es entstanden beispielsweise neue Handlungsbedarfe bzw. Stärken und Schwächen der vorhandenen Konzepte konnten aufgedeckt werden. Darüber hinaus konnten Teile des Informationsmodells für die Entwicklung neuer Sichten des IT-Managements eingesetzt werden (siehe Kapitel 6).

Um die Kosten zu bewerten, die bei der Entwicklung eines Informationsmodells anfallen, bewährte sich im Laufe dieser Arbeit der Grad an Modell-Veränderungen, der sich aufgrund neu aufgenommener Stakeholder Interessen ergibt. Beweist sich ein Modell nach mehreren iterativen Bearbeitungen als stabil, so ist daraus zu folgern, dass weitere Adressatenbefragungen nicht im Kosten-Nutzen-Verhältnis stehen würden.

Die Forderung nach der Persistenz des Informationsmodells, stellt Anforderungen auf die Verwendungsdauer des Modells ohne dessen Modifikation. Gemäß der Tatsache, dass grobe Modelle eine größere Persistenz besitzen als sehr detaillierte Modelle [Sc98], wurde bei der Entwicklung des Informationsmodells darauf geachtet, ein geeignetes Abstraktionsniveau zu finden. Einige Teilmodelle wurden recht abstrakt formuliert, um eine detaillierte Beschreibung der Sachverhalte in ergänzenden und den Einsatzszenarien gerecht werdenden Modellen zu ermöglichen (z.B. die Betriebs-Ebene). Es wurde stets darauf geachtet, geeignete Angriffspunkte für weiterführende Modelle zu liefern. So kann beispielsweise die Anwendungssystem-Ebene aufbauend auf der Klasse Softwareeinheit detaillierter in Form von Satzdefinitionen (HVB Systems spezifisches Datenformat), Datenbankeinträgen oder Klassen einer Programmiersprache beschrieben werden.

Lag jedoch der Bedarf der expliziten Darstellung detaillierter Sachverhalte, so wurden niedrigere Abstraktionsebenen gewählt. So beschreibt beispielsweise das Paket der Anwendunssystem-Ebene auf recht detaillierte Art und Weise, wie eine Anwendungslandschaft verwoben sein kann.

Zusätzlich zur Wahl des Abstraktionsgrades wurde weitestgehend auf moderne Begriffe verzichtet. Der Begriff Service wurde beispielsweise bewusst nicht für die Bennennung der von Anwendungssystemen bereitgestellten Funktionalitäten gewählt, da dieser Begriff derzeit sehr stark durch die (temporär aktuelle) serviceorientierte Architektur (SOA) verbreitet wird. Würde der Hype um die SOA verfliegen so würde das Modell durch den Begriff des Service an Aktualität verlieren. Die Wahl bzw. die Vermeidung moderner Begriffe unterliegt jedoch sehr stark subjektiver Wahrnehmung, so dass beispielsweise der Begriffe der Use-Case durch einen anderen Begriff ersetzt werden könnte.

Klarheit

Der Übersichtlichkeit und Strukturiertheit wird das Informationsmodell durch die Wahl der Modellierungssprache UML und der Trennung in Teilmodelle gerecht. Durch die Berücksichtigung graphische Anordnungsbeziehung, wie beispielsweise der maximalen Gradlinigkeit der Kanten, einer minimalen Kantenüberschneidung bzw. der Positionierung der Modellobjekte in möglichst gleichen Abständen und Ausrichtungen, wird die Übersichtlichkeit der Teilmodelle gefördert.

Der Nachteil des Modellierungswerkzeuges Rational Rose (IBM) lag in dem fehlenden Raster und verschiedenen Layoutalgorithmen, die eine automatische und optimale Ausrichtung der Objekte ermöglichen. Im Modellierungswerkzeug ARIS Webpublisher (IDS Scheer) werden beispielsweise vielfältigere Möglichkeiten angeboten.

Der Anordnungsregel von Becker, die Kantenziehungen nur in zwei orthogonale Dimensionen zu erlauben, wurde aus subjektiver Auffassung der Übersichtlichkeit nicht eingehalten.

Vergleichbarkeit

Die Vergleichbarkeit des Informationsmodells mit anderen Modellen ist aufgrund der Modellierungssprache UML und deren Transformierbarkeit gewährleistet. Ein Vergleich mit Modellen der Unternehmensarchitektur (vgl. 5.1) und speziellen Objektmodellen wurde im Laufe der textuellen Beschreibung des Modells vorgenommen (siehe 5.2).

Systematischer Aufbau

Das Informationsmodell wurde zwecks des Grundsatzes des systematischen Aufbaus in mehrere Teilmodelle zerlegt, so dass durch die Gesamtheit der Teilmodelle sichtenübergreifende Aspekte berücksichtigt werden konnten, ohne dem Grundsatz der Klarheit zu widersprechen. Teilmodelle der unterschiedlichen Ebenen werden dabei über die erneute Platzierung von Objekte integriert (vgl. Abbildung 39). Informationsobjekte, die in mehreren Paketen Verwendung finden, werden stets konsistent verwendet. Teilmodelle, wie beispielsweise das des IT-Management und der Betriebs-Ebene, setzten darüber hinaus Objekte anderer Ebenen unter neuen Aspekten in Beziehung.

Im Bezug auf den systematischen Aufbau ist jedoch anzumerken, dass das Ziel des entwickelte Informationsmodell lediglich ein Strukturmodell der Unternehmensarchitektur beinhaltet und keine verhaltensorientierte Beschreibungen umfassen soll. Die textuelle Beschreibung des IT-Management Modells gibt jedoch einen möglichen Hinweis für die Erstellung eines Verhaltensmodells.

Konstruktionsadäquanz

Um der Forderung des Konsenses zwischen Modellersteller und Adressat gerecht zu werden, wurde das Informationsmodell im ständigen Dialog mit den Modellnutzern entwickelt, verfeinert und validiert. Welche Objekte der Realität in das Informationsmodell aufgenommen werden sollten und welche nicht, wurde anhand des konstruierenden Problems und den in der Praxis existierenden Modellen abgewogen.

Sprachadäquanz

Aufgrund der Wahl der (semiformalen) Modellierungssprache UML wird der Grundsatz der Sprachadäquanz erfüllt. UML besitzt für die Modellierung des Informationsmodells eine angemessene semantische Mächtigkeit, da während der Modellierung, alle Sachverhalte des zu konstruierenden Problems, mittels zur Verfügung stehenden Modellierungsmitteln möglich war. Auch der Formalisierungsgrad der

UML Klassendiagramme bestätigte sich in den zahlreichen Diskussionen über und mit dem Modell. Selbst Personen, die bislang nicht mit UML Klassendiagrammen konfrontiert wurden, verstanden bereits nach wenigen Minuten die Inhalte der Teilmodelle und konnten über sie diskutieren.

Durch die zusätzliche schriftliche Dokumentation und Definition der Objekte ist eine exakte und verständliche Beschreibung der Sachverhalte möglich. Eine rein auf der natürlichen Sprache basierende Modellierung hätte die eindeutige Identifizierung von Modellobjekten und Modellbeziehungen erschwert und eine abstrakte Veranschaulichung der Sachverhalte nicht ermöglicht. Die Verständlichkeit und semantische Mächtigkeit natürlicher Sprachen erlaubt zwar eine schnelle Modellierung von Sachverhalten, erfordern aber beispielsweise bei einer Implementierung aufgrund des geringen Formalisierungsgrades zusätzliche Angaben und damit zusätzlichen Aufwand [BHZ04]. Ist eine Sprache dahingegen sehr formal, wie beispielsweise reguläre Ausdrücke, ist ein schnelles Verständnis für ungeschulte Personen unmöglich.

Die Wahl des Modellierungswerkzeuges Rational Rose (IBM) senkte dabei den Aufwand für die Modellerstellung beispielsweise aufgrund der Repository Unterstützung. Zusätzlich besteht der Vorteil von UML in Kombination mit Rational Rose (IBM) darin, automatisch beispielsweise Datenbank Tabellen oder Code einer speziellen Programmiersprache zu generieren.

5.4 Zusammenfassung der Modellbildung

Die entscheidende Leistung der Informationsmodellentwicklung besteht in der Identifikation und Beschreibung relevanter Objekte der Unternehmensarchitektur, die eine abstrakte Betrachtung der unterschiedlichen Architekturebenen ermöglichen und gleichzeitig den Zusammenhang der Ebenen durch ebenenübergreifende Beziehungen schaffen (vgl. [Bo05]). Abbildung 39 veranschaulicht die Beziehungen zwischen den Paketen, wobei sich die Abhängigkeiten auf importierte Klassen beziehen. Darüber hinaus können Beziehungen über die Datentypen von Attributen existieren.

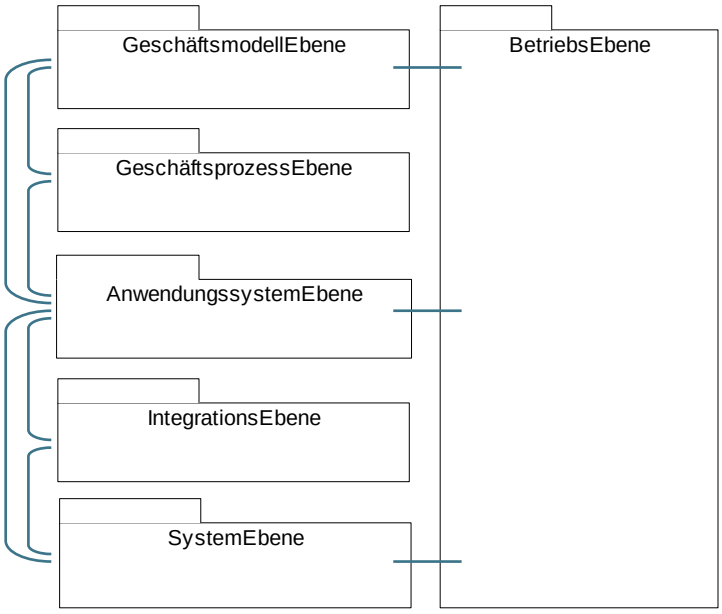


Abbildung 39: Beziehungen der Informationsmodell-Teilpakete

Das Informationsmodell liefert die Grundlage für handhabbare Sichten, die den Interessen und Fragestellungen unterschiedlicher Stakeholder gerecht werden. Das Verständnis über die Bestandteile und Zusammenhänge der Unternehmensarchitektur wird erleichtert, indem die Komplexität reduziert und eine Kommunikation ermöglicht wird.

Indem komplexe Sachverhalte spezifischer Teilbereiche des Unternehmens abstrahiert werden, können sie in den Gesamtzusammenhang des Unternehmens gebracht werden, so dass über die informationstechnische Kommunikation hinaus, eine Kopplung von Geschäft und IT gelingt.

Die Vorteile, die durch die Modellierung des Informationsmodells für das IT-Management geschaffen wurden, sind wie folgt zusammenzufassen (vgl. [Be00]):

1. *Beherrschung der Komplexität der Unternehmensarchitektur*: Die Abbildung relevanter Objekte der Unternehmensarchitektur in Form eines Informationsmodells, macht das in den einzelnen Bereichen enthaltene Wissen transparent. Der Brückenschlag von Geschäft und IT wird gefördert.
2. *Analyse der Unternehmensarchitektur*: Die Definition von Beziehungen zwischen Objekten einschließlich der Kardinalitäten ermöglichen das Bewusstsein der Zusammenhänge und die Identifikation von Handlungsbedarfen der Ist-Landschaft. Ein Baustein sollte beispielsweise genau einer Ebene zugeordnet sein, die Kardinalität im Informationsmodell zeigt jedoch, dass die Realität die Ist-Situation nicht dem gewünschten Soll entspricht.
3. *Gestaltung abteilungs- bzw. fächerübergreifender Kooperation*: Eine übergreifende Kooperation setzt die Definition von klaren Zuständigkeitsbereichen und Schnittstellen voraus, um eine klare Aufgabenverteilung zu ermöglich. Die Identifikation dieser Bereiche und Schnittstellen wird durch das Informationsmodell gefördert.
4. *Explikation des Wissens zum Zweck des Wissensaustauschs*: Die Voraussetzung für ein dezidiertes Wissensmanagement ist die Explikation des Wissens. Das Informationsmodell kann als ein vielfältig einsetzbares Dokumentationsmedium genutzt werden, das das betriebswirtschaftliche Wissen eines Unternehmens für unterschiedliche Zwecke speichert bzw. den Rahmen für weitere Ausarbeitungen bereitstellt.
5. *Mehrwert für spezielle Personengruppen*: Datenverarbeitungsabteilungen, als klassische Anwender von Informationsmodellen, können das Informationsmodell aufgrund seiner Methodenkompetenz beispielsweise für die Entwicklung von Datenmodellen verwenden. Darüber hinaus können Personen aus den Fachabteilungen oder den Organisationsabteilungen das Modell für die intuitive Nachvollziehbarkeit des mit der Modellsprache deklarierten Wissens nutzen. Nicht zuletzt kann das entwickelte Informationsmodell als Referenzmodell für die Erstellung von Modellen anderer Firmen eingesetzt werden.

Die Nutzung und Einhaltung von Grundsätzen ordnungsmäßiger Modellierung fördert darüber hinaus eine optimierte Darstellung und Qualität des Informationsmodells, so dass es ein probates Mittel darstellt, um die komplexe Unternehmensarchitektur beherrschen zu können.

Die abstrakte Darstellung der Objekte wurde von vielen Mitarbeitern der HVB Systems befürwortet und als hilfreich bzw. sinnvoll empfunden, da es bereichsübergreifende komplexe Sachverhalte und Zusammenhänge verdeutlicht. In einigen Fällen gab das Informationsmodell den Anstoß für die Hinterfragung bestehender Konzepte aus einer anderen Perspektive.

Das Informationsmodell soll nicht nur ein Dokument bzw. ein Stück Papier bleiben. Welche Möglichkeiten sich für die Weiterentwicklung bzw. Realisierung von Teilmodellen ergeben, veranschaulicht das folgende Kapitel beispielhaft ein zwei Szenarien. Grundsätzlich liegt die Verwendung des Informationsmodells in den Händen der HVB Systems Architekturabteilung. Diese Abteilung hat einen übergeordneten Charakter und muss als Dienstleister für die einzelnen Fachbereiche Konzepte und Methoden zu Verfügung stellen, um die geordnete Weiterentwicklung der Anwendungslandschaft und die einzelnen Projekte zu unterstützen. Als flankierende Maßnahme kann das erarbeitete Informationsmodell gelebt werden, indem Ansprechpartner für Teilmodelle identifiziert werden, die in ihren Zuständigkeitsbereichen das Modell publizieren und mit Daten füllen.

Kapitel 6

Umsetzung von Teilaspekten des Informationsmodells

Nachdem die Analyse bestehender Diagramme und Karten der HVB Systems in Kapitel 4 und das darauf aufsetzende Design des integrierten Informationsmodells in Kapitel 5 vorgestellt wurde, sollen im folgenden Kapitel zwei Teilmodelle implementiert werden. Es handelt sich dabei um das Modell der Anwendungssystem-Ebene, dessen Entstehung in Abschnitt 6.1 näher beschrieben und exemplarisch visualisiert wird, sowie dem Modell des IT-Management-Paketes, aus dessen Teilbereichen der so genannten IT-Masterplan entsteht (siehe Abschnitt 6.2).

6.1 Sichten für die Beschreibung der Anwendungslandschaft

In Abschnitt 5.2.1.3 wurde bereits das Teilmodell der Anwendungssystem-Ebene eingeführt. Da die Entwicklung und Abstimmung des Modells mehrerer iterativer Durchläufe bedurfte, sollen in diesem Abschnitt Anforderungen und Erfahrungen aus den Gesprächen mit den Mitarbeitern der HVB Systems aufgeführt werden. Ein Ansatz für die Betrachtung der Anwendungslandschaft auf unterschiedlichen Detailebenen ergänzt dabei die Ausführung.

6.1.1 Anforderungsanalyse

Zahlreiche vergangene und aktuelle Projekte der HVB Systems zeigen, dass der Bedarf an einer geeigneten Beschreibung der Anwendungslandschaften einschließlich der Vernetzung beteiligter Geschäftsprozesse und Anwendungssysteme stetig wächst. Unterschiedliche Hintergründe und Interessen führten dabei bislang insbesondere bei der Beschreibung von Schnittstellen zu differierenden Darstellungen. Kostspielig erhobene Daten wurden nur innerhalb des zugehörigen Projektes verwendet und wurden später zu Datenmüll.

Um langfristig eine einheitliche und bereichsübergreifende Bereitstellung von Informationen über die Zusammenhänge von Anwendungssystemen, einschließlich den von ihnen unterstützten Geschäftsprozessen zu gewährleisten, entstand die Forderung nach geeigneten Sichten, die eine zukünftige Datenerhebungen nach einem einheitlichen Schema vorschreiben. Daten sollen zentral abgelegt und verwaltet werden, um für spätere Anwendungsfälle einheitlich bereitgestellt zu werden.

Folgende vier Szenarien sollen den Einsatz dieser Sichten und die Verwendung der bereitgestellten Daten verdeutlichen:

1. *Erstellung neuer Schnittstellen*: Werden neue Schnittstellen entwickelt, muss überprüft werden können, ob bereits eine passende Schnittstelle existiert bzw. entsprechend erweitert werden kann. Die Suche gleicher oder ähnlicher Schnittstellen soll ermöglicht bzw. die Entwicklung doppelter Schnittstellen vermieden werden [HVB05f].
2. *Einführung einer serviceorientierten Architektur*: Erfolgt eine Migration der jetzigen Ist-Anwendungslandschaft zu einer serviceorientierten Architektur werden Informationen benötigt, die die Identifikation von Enterprise-Services unterstützen.
3. *Identifikation von Handlungsbedarfen*: Für den Fall, das Anwendungssysteme einer Änderung, Ablösung oder einem Ausfall unterliegen, müssen bereits im Vorfeld Handlungsbedarfe aufgedeckt werden können. Hierzu werden Informationen über die Vernetzung der Anwendungssysteme, sowie deren Lebenszyklen benötigt, um Abhängigkeiten zu identifizieren.
4. *Berechnung von Geschäftsprozesskosten und -durchlaufzeiten*: Die gesamten Kosten bzw. die Durchlaufzeiten für die Durchführung eines Geschäftsprozesses inklusive der Kosten und Durchlaufzeiten die für die unterstützenden Anwendungssysteme anfallen, müssen berechnet werden können. Die durchgängige Dokumentation der dafür benötigten Sachverhalte kann dazu dienen, Handlungsbedarfe für Geschäftsprozessoptimierungen aufzudecken und Medienbrüche zu vermeiden.
5. *Testen von Teilsystemen*: Bislang beschränkt sich das Testen neu entwickelter Teilsysteme auf die wichtigsten technischen und funktionalen Aspekte des Systems. Es ist jedoch wün-

schenswert, vor der Einführung der Teilsysteme dessen Funktionalität in Bezug auf die von ihnen unterstützten Geschäftsprozesse zu überprüfen. Systemtests sollen ausführlicher und teilweise automatisiert werden, indem beispielsweise an Schnittstellen von PASS bzw. ARIS Toolset (IDS Scheer) angesetzt wird, um Informationen über Geschäftsprozesse sowie den zugehörigen Teilsystemen auszulesen und entsprechende Test zu generieren. Das bisherige Testen der reinen Funktionalität eines Teilsystems würde sich zu einem Testen von Geschäftsprozessen entwickeln.

Die Szenarien zeigen, dass die Sichten zur Beschreibung der Anwendungslandschaft Interessen und Fragestellungen vielfältiger Stakeholder berücksichtigen müssen. Es muss sowohl eine Beschreibung auf grober als auch auf feiner Ebene ermöglicht werden, sowie die Bereitstellung bereichsübergreifender Informationen, die sich von den Geschäftsprozessen über die Anwendungssysteme bis hin zu den Softwarekomponenten erstrecken. Eine durchgängige Integration der fachlichen und technischen Informationen ist dabei eine zwingende Voraussetzung. Es muss beispielsweise möglich sein, bei einem Übergang zweier Teilprozesse festzustellen, ob auch ein Übergang bei den unterstützenden Anwendungssystemen stattfindet.

6.1.2 Entwicklung der Sichten

Grundlage der Sichten ist das in Abschnitt 5.2.1.3 eingeführte Informationsmodel, das die Komponenten der Anwendungslandschaft abbildet. Um der Forderung unterschiedlicher Abstraktionsgrade gerecht zu werden, ist es möglich, je nach Anforderungen nur Teile des Informationsmodells zu betrachten, für die die entsprechenden Daten erhoben wurden. Aufgrund der im Informationsmodell gewählten Kardinalitäten, können Verbindungen sowohl auf grobgranularer Ebene, d.h. Verbindungen von Anwendungssystemen im Zusammenhang mit Geschäftsprozessen, als auch auf feingranularer Ebene, d.h. Verbindungen von Komponenten der Teilsysteme einschließlich der Konnektoren beschrieben werden (vgl. [MW04b]).

Folgend sollen einzelne Detailebenen einschließlich beispielhafter graphischer Umsetzungen beschrieben werden, die die Betrachtung der Sachverhalte von unterschiedlichen Abstraktionsebenen ermöglichen.

6.1.2.1 Detailebenen

Auf der untersten Ebene (*Detailebene 0*) werden Anwendungssysteme in den Zusammenhang mit Organisationseinheiten, Building Blocks und Geschäftsprozessen gebracht. Bei der Visualisierung dieser Ebene erschließen sich die Abhängigkeiten beteiligter Objekte aus ihrer Position (siehe Abbildung 40).

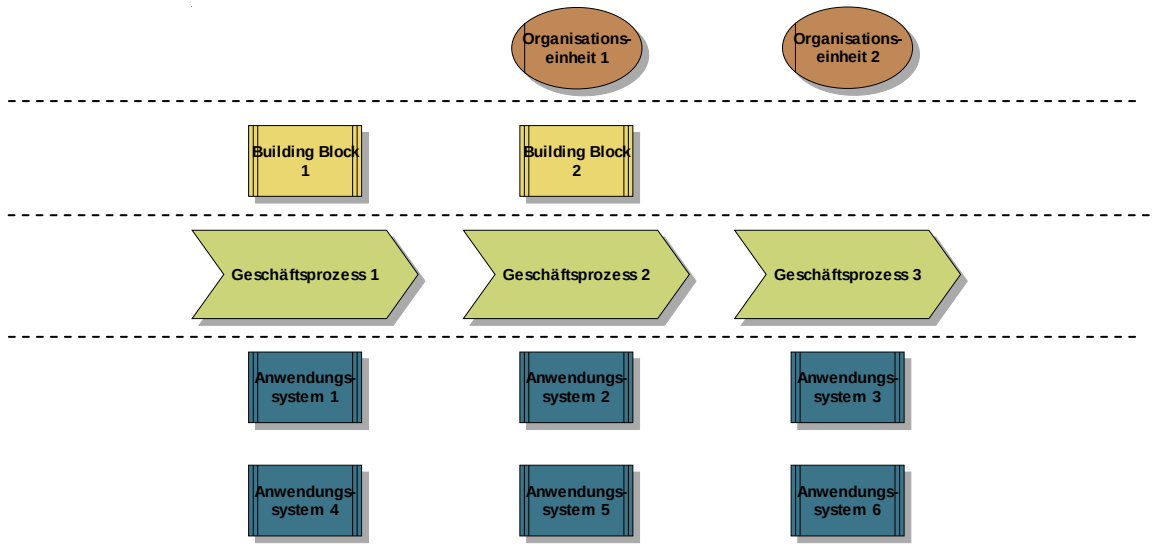


Abbildung 40: Visualisierung von Verbindungen auf Detailebene 0

In *Detailebene 1* werden existierende Verbindungen zwischen Anwendungssystemen erfasst und durch Linien veranschaulicht (siehe Abbildung 41). Um die Verbindung mittels Attributen näher zu spezifizieren, wird auf der Verbindungslinie ein Knoten dargestellt, der bei Doppelklick eine Ansicht auf die Attribute öffnet.

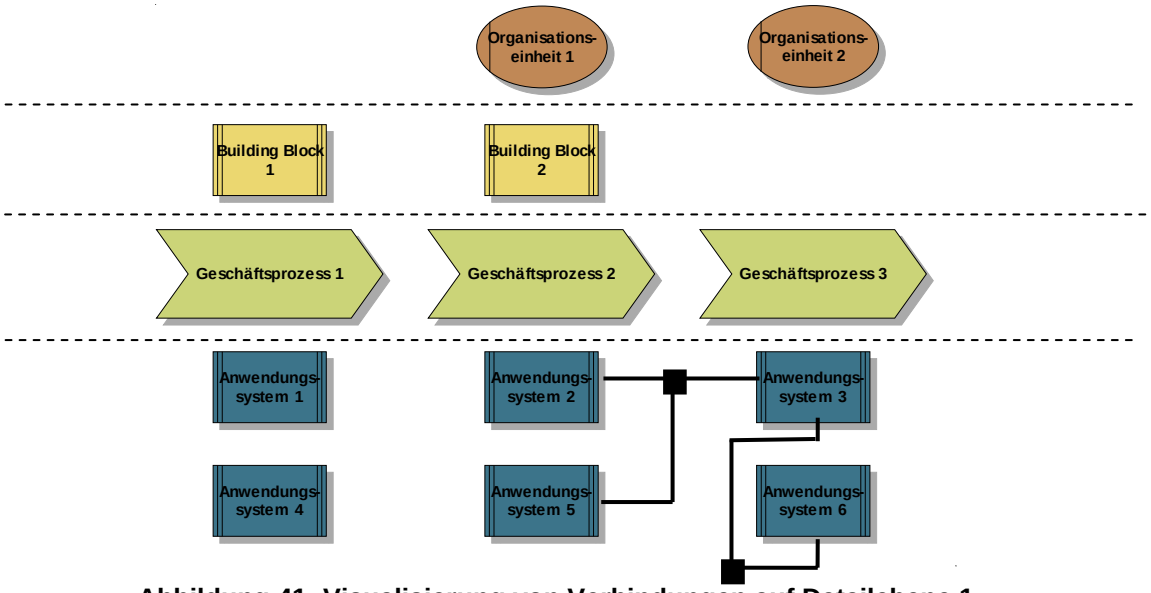


Abbildung 41: Visualisierung von Verbindungen auf Detailebene 1

Der Informationsgehalt der Detailebene 1 soll den Anforderungen der Fachbereiche dienen. Aus diesem Grund werden gleichartige Verbindungen zu mehrfach Beziehungen aggregiert. Tragen beispielsweise Anwendungssystem 2 und 5 dazu bei, Daten für den nächsten Prozessschritt bereitzustellen, der von Anwendungssystem 3 unterstützt wird, werden die zwei Verbindungen zu einer Verbindung mit drei Teilnehmern zusammengefasst.

Interessiert sich ein Stakeholder dennoch für eine genauere Beschreibung der Verbindung, so soll durch den Doppelklick auf das Verbindungssymbol eine ausführlichere Ansicht (in Toolset ARIS von IDS Scheer z.B. Zugriffsdiagramm) geöffnet werden. Eventuell kann zuvor eine Auswahl der gewünschten Beziehung angeboten werden, für die in einer neuen Ansicht gerichtete Pfeile die Richtung des Datenflusses oder der Funktionsnutzung (Anbieters, Nutzer) darstellen. Unterschiedliche Pfeiltypen erlauben dabei die Unterscheidung zwischen Datenfluss und Funktionsnutzung (vgl. Abbildung 42).

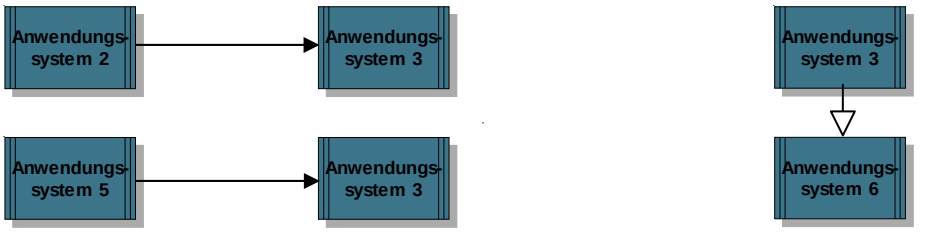


Abbildung 42: Detailebene 1 (ausführlich): links Datenfluss, rechts Funktionsnutzung

Detailebene 2 zeichnet sich durch die Betrachtung von Teilsystemen aus, die entsprechend den vorherigen Ebenen zusammen mit Geschäftsprozessen, Organisationseinheiten und Building Blocks abgebildet werden (siehe Abbildung 43). Wie auf Detailebene 1 wird die Verbindung zweier Teilsysteme mittels Linien dargestellt und kann nach Doppelkick auf den Knoten durch weitere Attribute oder der ausführlichen Sicht vertieft werden. Die Praxis zeigte, dass es in einigen Fällen nicht möglich ist, die beteiligten Teilsysteme zu identifizieren, so dass anstelle dessen die Anwendungssysteme aufgeführt werden müssen. Zudem sei darauf hingewiesen, dass auf dieser Detailebene Verbindungen sichtbar werden können, die in Detailebene 1 noch nicht zu sehen waren. Es handelt sich dabei um

Verbindungen die zwischen Teilsystemen *eines* Anwendungssystems existieren (vgl. die Teilsysteme 1 und 2 des Anwendungssystems 2 in Abbildung 43).

An dieser Stelle soll die Visualisierung der Detailebenen einem Kartentyp der Softwarekartographie zugeordnet werden. Bei den Darstellungen handelt es sich um Matrixkarten, die jedoch den Prozessunterstützungskarten sehr stark ähneln, da die enthaltenen Elemente anhand von Geschäftsprozessen verortet werden. Der Unterschied zu den Prozessunterstützungskarten liegt jedoch darin, dass die Geschäftsprozesse nicht in der ersten Zeile der Matrix erscheinen, sondern erst in der dritten. Die erste Zeile umfasst die Organisationseinheiten und die zweite Zeile die Building Blocks, die einem Geschäftsprozess zugeordnet werden können. Die Zeilen unterhalb der Geschäftsprozesse beinhalten die Anwendungs- bzw. Teilsysteme, die einen Geschäftsprozess unterstützen. Die Geschäftsprozesse wurden somit zwischen den restlichen Zeilen platziert, um die Zugehörigkeitsbeziehungen deutlicher darzustellen. Würden sie in der ersten Zeile aufgeführt, so könnte der Kartennutzer die Anwendungs- und Teilsysteme fälschlicherweise den Building Blocks oder den Organisationseinheiten zuordnen.

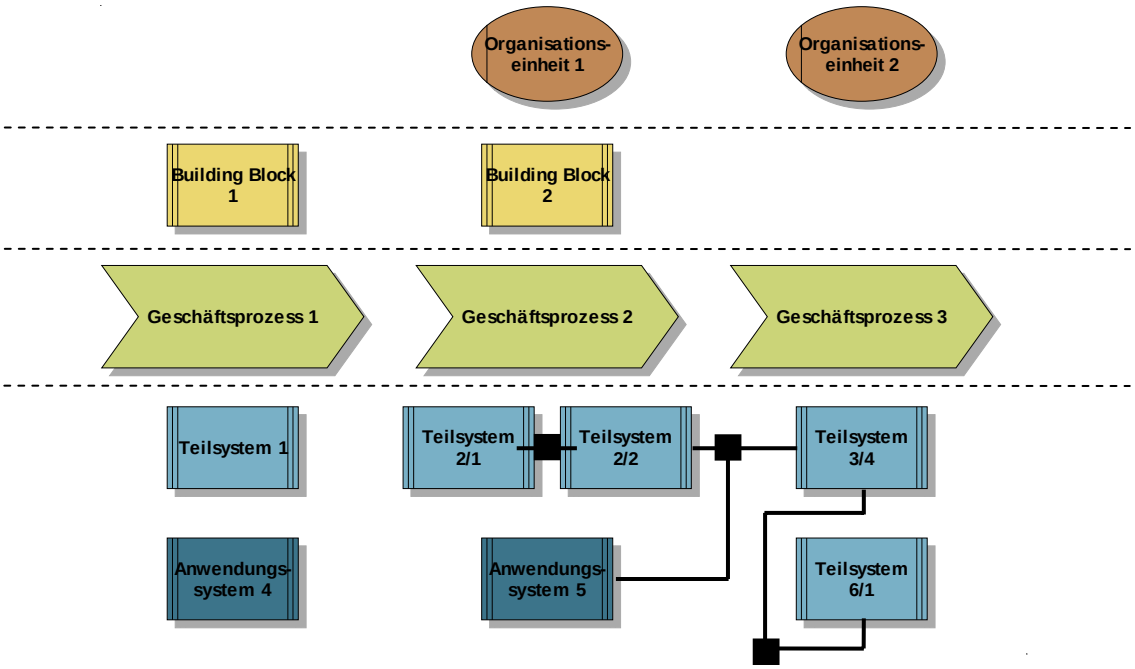


Abbildung 43: Visualisierung von Verbindungen auf Detailebene 2

Die ausführlichste Betrachtung von Schnittstellen wird in der *Detailebene 3* vorgenommen, in der zwischen einer fachlichen und einer technischen Ansicht unterschieden wird. In der fachlichen Ansicht werden Anwendungssysteme mit ihren Teilsystemen und den von den Teilsystemen realisierten Use-Cases in Zusammenhang gebracht. Informationen über die Verbindung können mit Hilfe fachlicher Export- und Import-Konnektoren dokumentiert werden (siehe Abbildung 44). Die technische Ansicht zerlegt dahingegen Teilsysteme in Software-Komponenten, die für die Umsetzung der eigentlichen Schnittstelle (Verbindung) verantwortlich sind. Fachliche und technische Export- bzw. Import-Konnektoren liefern darüber hinaus detaillierte Angaben über die Schnittstelle und deren Realisierung bzw. Nutzung. Linien veranschaulichen in dieser Ansicht Verbindung von Software-Elementen, bzw. falls die Informationen nicht vorhanden sind von Software-Komponenten (siehe Abbildung 45).

Für die Visualisierung der Export-Konnektoren bietet sich dabei das von vielen Visualisierungssprachen (z.B. der UML) bekannte Lollipop-Symbol an. Import-Konnektoren können durch ein ausgefülltes Rechteck auf der Seite des Anbieters dargestellt werden.

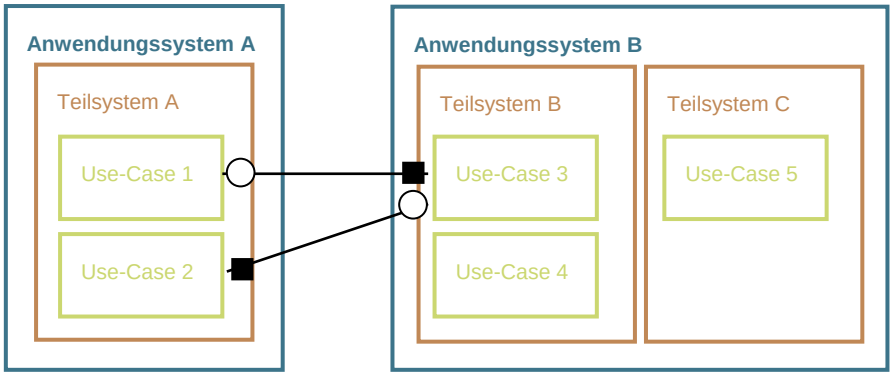


Abbildung 44: Fachliche Verbindungen auf Detailebene 3

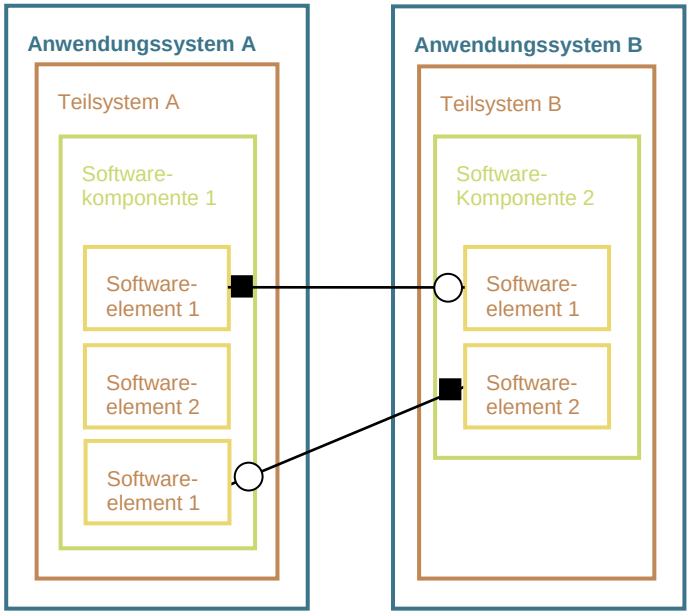


Abbildung 45: Technische Verbindungen auf Detailebene 3

Die Informationen der Detailebene 3 müssen nicht zwingend visualisiert werden, da eine graphische Darstellung der gesamten Anwendungslandschaft zu unübersichtlich wird. Die Daten sollten im Firmen-Repository abgelegt und beispielsweise durch weiterführende Links auf Detailebene 2 angeboten werden.

Zwei weitere Ansichten können die zuvor beschriebenen Detailebenen ergänzen. Bei der Ersten handelt es sich um die Darstellung von Datenflüssen, die entweder zwischen den (manuellen) Geschäftsprozessen oder den Anwendungs- bzw. Teilsystemen stattfinden können. Bei der Erfassung der Datenflüsse ist es vorstellbar, eine eigene Hierarchie zu bilden. Auf einer hohen Abstraktionsebene können beispielsweise nur die Informationsträger von Bedeutung sein, wie beispielsweise Fax oder E-Mail. Auf niedrigeren Abstraktionsebenen sind dahingegen die konkreten Übertragungsobjekte von Interesse, wie beispielsweise eine konkrete E-Mail.

Die zweite ergänzende Ansicht soll bei einem Doppelklick auf ein Anwendungs- bzw. Teilsystem alle zugehörigen vorgelagerten (System nimmt Daten/ Funktionsaufruf entgegen) und nachgeschalteten Systeme (System nutzt Daten bzw. stößt Funktionsaufruf an) auflisten (siehe Abbildung 46).

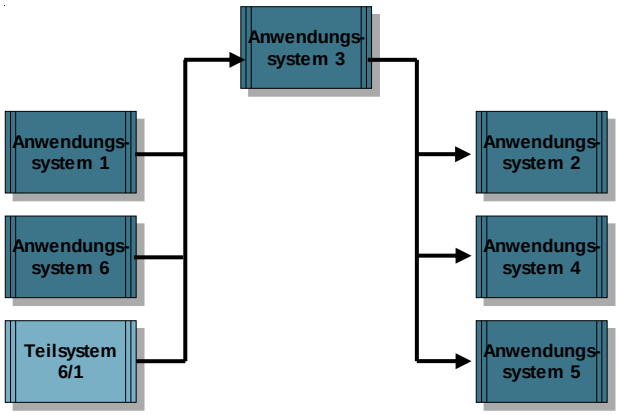


Abbildung 46: Vorgelagerte und nachgeschaltete Systeme

6.1.2.2 Problematik der Datenbeschaffung

Bei der Entwicklung des Informationsmodells wurde bereits deutlich, dass bei einer Umsetzung in der HVB Systems die (initiale) Datenbeschaffung problematisch sein wird, da bislang nicht alle benötigten Daten vorhanden sind. Informationen für die Detailebene 0 liegen größten Teils in PASS vor. Das Zusammenspiel von Anwendungssystemen und Geschäftsprozessen, wie es für die Detailebene 1 benötigt wird, muss bisher aus unterschiedlichen Informationsquellen zusammengetragen werden. Welche Teilsysteme zu einem Anwendungssystem gehören, ist bereits Teil des Repositorys, die von einem Teilsystem realisierten Geschäftsfunktionen müssen jedoch für die Detailebene 2 identifiziert und spezifiziert werden. Für Detailebene 3 müssen sämtliche Informationen über Export- und Import-Konnektoren erhoben und im Firmen-Repository abgelegt werden.

Ein möglicher Ansatz für die Beschaffung der restlichen Daten müsste sowohl *top-down* als auch *bottom-up* vollzogen werden. In dem *top-down* Ansatz müssten die Informationen über die Vernetzung der Anwendungslandschaft mit Hilfe der Fachbereiche gesammelt werden. Mit Hilfe einer Umfrage, die beispielsweise durch Verschicken eines Fragebogens durchgeführt werden könnte, könnten Informationen über Anwendungssysteme, Use-Cases sowie fachliche und technische Konnektoren erhoben werden. Der Nachteil dieses Verfahrens liegt jedoch zum einen darin, dass die einmalig erhobenen Daten veralten. Zum anderen kann die Auswertung der Umfrage falsche technische Informationen enthalten. Eine Person kann beispielsweise die Aussage treffen: "Es gibt eine Schnittstelle mit der Funktion Kundendaten holen", es ist jedoch noch nicht gewährleistet, dass ein entsprechender technischer Konnektor tatsächlich existiert, der diese Funktion anbietet. Aus diesem Grund muss das *top-down* Verfahren durch einen *bottom-up* Ansatz ergänzt werden. Überwiegend technische Informationen sollen mittels Parsen von Programmcode ermittelt werden. Der Nachteil dieses Ansatzes besteht jedoch darin, dass die anfallende Datenmenge sehr groß und in ihrer Auswertungsqualität eingeschränkt ist. Es stellt sich die Frage, wie Informationen über Methodenaufrufe, Parameter, Portangaben etc. den fachlichen Aussagen zugeordnet werden können. Es steht fest, dass bislang keine Informationen über die fachlichen Aspekte im Code verankert wurden, so dass der Abgleich von technischen und fachlichen Informationen beider Ansätze per Hand durchgeführt werden muss. Die Zuordnung muss dabei von den Verantwortlichen der Anwendungssysteme bzw. den Verantwortlichen der zugehörigen Building Blocks durchgeführt und angestoßen werden. Als Hilfestellung können Dokumentationen der Anwendungssystem-Entwicklung, wie beispielsweise ein fachliches Klassenmodell, Sequenzdiagramme oder Programmablaufdiagramme (ARIS Toolset von IDS Scheer) in Zusammenhang mit EPKs dienen.

Des Weiteren wurde festgestellt, dass die Datenbeschaffung für Host (zentralen) Anwendungen einfacher durchzuführen ist, als die der dezentralen Anwendungen. Zu begründen ist dies mit dem "Wildwuchs" von dezentralen Anwendungen, hervorgerufen durch unterschiedlichste Technologien (Struts, EJB etc.). Zukünftig sollen neue Technologien und Methoden wie beispielsweise der *Deployment Deskriptor* der EJBs oder der Deskriptor von Microsoft im Zusammenhang mit *Assembly* und .NET helfen, Informationen über die beteiligten Komponenten und Schnittstellen eines dezentralen Systems bereitzustellen. Mit einer Sprache wie beispielsweise der *Business Process Execution Language for*

Web Services (BPEL4WS, Microsoft und IBM) soll dagegen das Zusammenwirken unterschiedlicher Geschäftsprozesse beschrieben werden.

Um die kontinuierliche Datenpflege insbesondere bei neu entwickelten Systemen zu gewährleisten, kann ein einheitliches Vorgehen in Projekten z.B. über das Vorgehensmodell ALADIN adressiert werden.

6.1.3 Abschließende Bemerkungen

Der Bedarf an den Informationen rund um die Anwendungslandschaft steigt, wobei der Fokus derzeit stark auf der Vernetzung der Anwendungssysteme und die von ihnen unterstützenden Geschäftsprozesse liegt. Diese Informationen werden insbesondere auf der Managementebene benötigt, um die Weiterentwicklung der Anwendungslandschaft geeignet steuern zu können.

Der Abschnitt hat aufbauend auf dem Teilmodell des Anwendungssystem-Ebenen Paketes mögliche Detailebenen für eine Beschreibung der Anwendungslandschaft aufgeführt, auf die Problematiken der Datenbeschaffung hingewiesen und erste Ansäte für eine graphische Repräsentation geliefert.

6.2 IT-Masterplan

Mit zu den wichtigsten Teilaufgaben des IT-Managements zählen die Planung und die Steuerung von IT-Projekten. Um Informationen der geplanten und laufenden Projekte geeignet erfassen und verstehen zu können, bedarf es eines Hilfsmittels wie beispielsweise einem so genannten IT-Masterplan, der die Zusammenhänge des fachlichen und des technischen Portfolioplans in Form einer graphischen Repräsentation veranschaulicht. In den folgenden Abschnitten soll der IT-Masterplan erläutert und dessen Entwicklung von der Anforderungsanalyse über die Wahl der kartographischen Mittel bis hin zu der fertigen Softwarekarte in Abschnitt 6.2.3 erläutert und beschrieben werden.

6.2.1 Anforderungsanalyse

Die Architekturabteilung der HVB Systems hat unter anderem die Aufgabe, Fachbereiche und deren Prozesse zu unterstützen, neue IT-Produkte bzw. Methoden zu bewerten und Projektunterstützung zu gewährleisten. Um diesen Aufgaben gerecht zu werden, müssen Bedarfe der Fachbereiche frühzeitig erkannt werden, um zum erforderlichen Zeitpunkt benötigte Infrastrukturen inklusive Plattformen bereitstellen zu können. Aufgrund der zentralen Rolle der Abteilung muss sie den Überblick über laufende Projekte besitzen, um Abhängigkeiten von Projekten zu identifizieren und geeignet reagieren zu können.

Die Vergangenheit zeigte, dass die Architekturabteilung von den Anforderungen der Fachbereiche „überrascht“ wurde und mehr damit beschäftigt war einzelne Anforderungen nachträglich zu erfüllen, anstelle im Vorfeld eine zukunftsfähige Infrastruktur bereitstellen zu können. Um "das Hinterherhängen" der Infrastruktur in eine "Vorrausschauende" umzuwandeln, bedarf es einer Übersicht der Programme/Projekte aus den Fachbereichen, der Abstimmung mit den Plattformen und der Ableitung von Maßnahmen und Empfehlungen. Diese Übersicht soll im so genannten IT-Masterplan erfolgen, der im Intranet veröffentlicht werden, sowie den Fachbereichen, den Beteiligten von Programmen/Projekten und den Architekten, als ein Werkzeug für geeignete Projektsteuerung dienen soll.

Im IT-Masterplan werden Informationen des fachlichen Portfolioplans mit denen des technischen Portfolioplans zusammengeführt. Der fachliche Portfolioplan bringt zunächst die vom Fachbereich initiierten Programme und Projekte zeitlich in Zusammenhang, wohingegen der technische Portfolioplan alle Plattform[34] betreffenden Projekte umfasst. Laut CCTA (Central Computer and Telecommunications Agency) versteht man dabei unter einem Programm: “(…) portfolio of projects and changes, planned and managed in a coordinated way, that change organisations to achieve a set of defined business

[34] Technologische Ausprägung eines oder mehrerer Bausteine der Integrations-Ebene. Sie umfasst Produkte, Prozesse (z.B. Implementierung, Deployment) und steht im Zusammenhang mit einer bestimmten Infrastruktur und einer Umgebung (z.B. Entwicklungsumgebung).

objectives or benefits that are of strategic importance" [OGC05]. Ziel des IT-Masterplanes ist die Abstimmung des Softwarearchitektur-Bebauungsplans[35] mit relevanten Programmen und Projekten[36].

Abhängigkeiten zwischen den Projekten sollen verdeutlicht werden, um Maßnahmen und Empfehlungen ableiten zu können (vgl. [BES04]). Werden beispielsweise zwei Projekte auf einer Plattform durchgeführt, ist zu ermitteln, ob sie sich gegenseitig beeinträchtigen. Ebenso wichtig sind Informationen über den Zusammenhang von geplanten Projekten, die bestimmte Plattformen voraussetzten. Ist eine Plattform zu dem vom Projekt geforderten Zeitpunkt nicht verfügbar, müssen geeignete Maßnahmen getroffen werden. Aus dem IT-Masterplan nicht zu erkennende Abhängigkeiten sind solche, die die fachliche Überlappung von Projekten betrifft.

Aus der zeitlichen Abstimmung und dem Erkennen von Beziehungen und Abhängigkeiten zwischen Programmen/Projekten, sollen Maßnahmen und Empfehlungen abgeleitet werden können. Auf diese Weise gewonnene Kenntnisse können beispielsweise Auswirkungen auf den Softwarearchitektur-Bebauungsplan besitzen, in den eventuell neue Bausteine oder Dienste aufgenommen werden müssen. Neue Plattformprojekte können sich ergeben, bestehende können sich verschieben oder sogar wegfallen, falls sie nicht der IT-Strategie entsprechen. Ebenso sind als Auswirkung auch zeitliche Verschiebung eines Programmes/Projektes oder eines Plattformprojektes aus der Abstimmung mit dem Softwarearchitektur-Bebauungsplan vorstellbar.

Nachdem Motivation und Inhalt des IT-Masterplanes erläutert wurden, soll entsprechend dem in Kapitel 4 eingeführten Kartenanalyseschema, unterschiedliche Rollen einschließlich ihrer Interessen und Fragestellungen aufgeführt werden. Bei den Stakeholdern handelt es sich hierbei um weniger als tausend Personen, die entweder der Architekturabteilung, den Fachbereichen angehörig oder Beteiligte von konkreten Programmen/Projekten sind. Zu ihren Interessen zählt das Erkennen von Handlungsbedarf, um Programme und Projekte geeignet steuern zu können. Ihre Fragen zielen auf die Machbarkeit der Projekte bzw. der Zukunftsaussichten der Plattformen ab.

Die Rolle des Analysten und des *Modelers* nimmt zukünftig die Architekturabteilung ein, d.h. weniger als zehn Personen. Für den Analysten ist es entscheiden, welche aktuellen strategischen Programme/Projekte durchgeführt werden, in welcher Phase sich die Programme/Projekte befinden und welchen Status die aktuellen Plattformen besitzen.

Die Anzahl der *Information Supplier* ist erheblich größer, bleibt jedoch unter tausend Personen, die den Fachbereichen bzw. der Architekturabteilung angehörig oder Projekt bzw. Building Block Leiter sind. Die Datenbeschaffung selbst wurde händisch vorgenommen, da keine Datenbasis vorlag, die alle benötigten Informationen umfasste. Mit Hilfe von Dokumenten (z.B. erstellt mit Excel von Mikrosoft), die Informationen über Projekte in unterschiedlichen Abteilungen beinhalteten, dem Dokumentationswerkzeug ALADIN, Ordnern auf Projektlaufwerken, Intranet Recherche und persönlichem Kontakt mit den *Information Suppliern* mussten die Daten für den IT-Masterplan zusammengetragen werden. Bei der während dieser Arbeit erstellten Version wurden alle Projekte des Jahres 2005 der Architekturabteilung und eine Auswahl der aktuell bekannten Fachbereichsprojekte berücksichtigt. Zukünftig sollen auch Projekte der HVB Info in den IT-Masterplan einfließen, um eine optimale Abdeckung aller relevanten Teilbereiche zu gewährleisten.

Die aus der Analyse resultierenden relevanten Informationsobjekte werden in Abbildung 47 veranschaulicht. Auf die Darstellung der Attribute wurde aus Platzgründen verzichtet.

[35] Der Softwarearchitektur-Bebauungsplan sorgt für die Bereitstellung standardisierter Plattformen (siehe Anhang A).

[36] Vgl. strategische IT-Vorhaben, IT-Vorhaben des Geschäftsbereiches, Vorhaben der IT-Bereitstellung in [GA02].

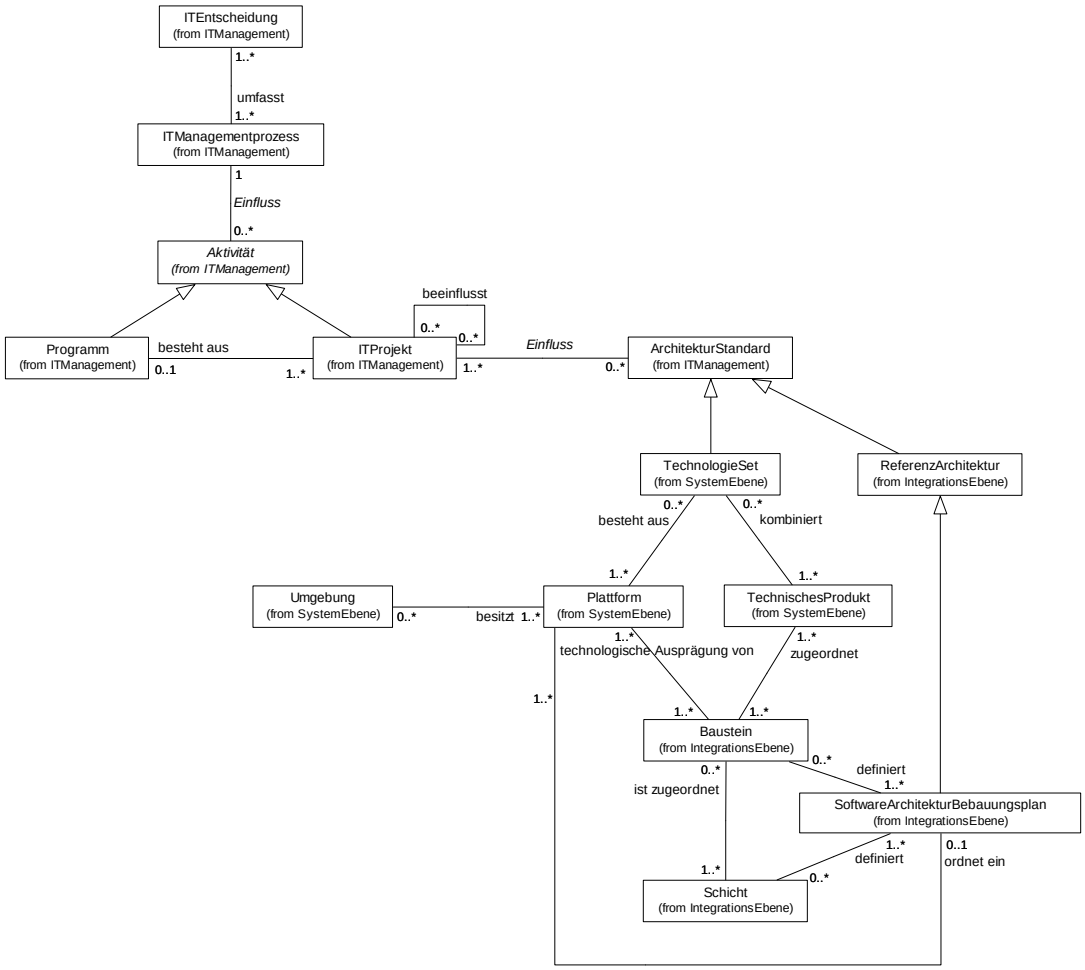


Abbildung 47: Informationsobjekte des IT-Masterplans

Programme bestehen aus mindestens einem IT-Projekt und sind aufgrund ihres Namens eindeutig identifizierbar. IT-Projekte können einem Programm zugeordnet sein. Da es sich bei einem IT-Projekt entweder um ein Projekt aus den Fachbereichen oder einem Plattformprojekt handeln kann, können sich IT-Projekte gegenseitig beeinflussen. Sowohl Programme, als auch IT-Projekte werden durch den (geplanten) Beginn bzw. ihr Ende und den (ALADIN) Meilensteinen charakterisiert. Mittels der Zeitplanung der IT-Projekt Meilensteine „Analyse“, „Design“, „Realisierung“ und „Einführung“, lassen sich die Abhängigkeiten zu Plattformprojekten genauer bestimmen, da beispielsweise eine Plattform erst nach dem Design zur Verfügung stehen muss und nicht schon mit Beginn eines Projektes. Die Meilensteine der Plattformprojekte sind „Analyse“, „Design“, „Realisierung“, „Einführung“ und „Produktiv“. Handelt es sich um ein Plattformprojekt, so hat dieses einen Einfluss auf IT-Architekturstandards, d.h. auf den Softwarearchitektur-Bebauungsplan oder den Technologiesets. Zusätzlich stehen Projekte unter ständigem Einfluss des Architekturmanagement-Prozesses, so dass IT-Entscheidungen den Verlauf von Projekten beeinflussen.

Für neu einzuführende Plattformen ist es notwendig, den Zeitplan für die Bereitstellung der einzelnen Umgebungen einer Plattform darzustellen. Hierbei handelt es sich um die Entwicklungsumgebung (EU), die Integrationstestumgebung (IT), die Qualitätssicherungsumgebung (QSU) und die Produktion.

Die mit den Plattformen in Verbindung stehenden Architekturstandards für die Softwarearchitektur, wie hier der Softwarearchitektur-Bebauungsplan, kategorisieren technische Produkte mittels definierter Schichten und Bausteine.

Die Klassen IT-Architekturstandard, Aktivität, Softwarearchitektur-Bebauungsplan und IT-Management Prozess sollen nicht auf dem IT-Masterplan visualisiert werden. Sie dienen lediglich der Veranschaulichung der oben beschriebenen Zusammenhänge. Technologiesets treten nur dann auf dem IT-Masterplan auf, wenn sie als Plattform in einem Projekt eingesetzt werden.

Im folgenden Abschnitt sollen Aufgaben und Techniken der Kartographie aufgeführt werden, die für die Realisierung des IT-Masterplanes von Bedeutung sind.

6.2.2 Grundlagen der Kartographie

In Kapitel 4.1 wurde die Kartengraphik als Zeichensystem mit den Gestaltungsmitteln und Gestaltungsvariablen eingeführt. In diesem Abschnitt sollen darauf aufbauend Rahmenbedingungen für die Anwendung der Kartengraphik aufgeführt werden, die bei der Realisierung des IT-Masterplanes zu berücksichtigen sind.

Zunächst führt die Bedeutung eines Kartenzeichens bei den Gestaltungsmitteln und ihrer Variation zu folgenden Grundsätzen [HGM02]:

- Gleiches gleich, Ungleiches ungleich darstellen
- Wichtiges erhalten, Unwichtiges fortlassen
- Typisches betonen, Untypisches abschwächen

Darüber hinaus setzt die Lesbarkeit der einzelnen Kartenzeichen eine visuell noch wahrnehmbare graphische Mindestgröße voraus, die vom menschlichen Sehvermögen und von der Leistungsfähigkeit der Kartentechnologie abhängt. Für Bildschirmkarten werden beispielsweise serifenlose Schriftarten mit einer Mindestschriftgröße von 12pt empfohlen. Des Weiteren ist der Kontrast einer Darstellung von entscheidender Bedeutung. Ein Buchstabe der in einer schwarz weiß Darstellung abgebildet wird hat beispielsweise laut Hake et al. [HGM02] eine Mindestgröße von 0,6 mm, wohingegen ein Buchstabe auf farbigem Grund eine Mindestgröße von 1 mm aufweisen sollte.

Die Lesbarkeit und das Verständnis von Karten werden zudem gefördert, wenn die Wahrnehmbarkeit der typischen Gestalt des Kartenzeichens gewährleist wird und die Realisierbarkeit bzw. Konstanthaltung des Zeichens in technischen Prozessen wie Zeichnung oder Vervielfältigung sichergestellt wird [HGM02].

Letzte aber dennoch sehr wichtige Erkenntnis beschäftigt sich mit der Gestaltungswahrnehmung. Das Prägnanzprinzip (Prinzip der guten Gestalt) bewirkt, dass derjenige Zusammenschluss von Elementen bevorzugt wird, der eine möglichst geschlossene, stabile, in sich folgerichtige und einfache Gestalt ergibt. Für die Kartengraphik ergeben sich hieraus einige Regeln. Zunächst muss die graphische Differenzierung ausreichend sein. Hilfsmittel sind hier die unterschiedlichen Gestaltungsmittel unter Einsatz verschiedener Gestaltungsvariablen. Des Weiteren darf die graphische Dichte nicht zu groß sein, wobei sich die Dichte durch die Kartengröße (Höhe, Breite) oder durch die Nutzung des Schichten- und *Viewpoint*-Prinzips beeinflussen lässt. Außerdem müssen Kontrast und Objekttrennung ausreichend sein, indem beispielsweise ein heller Untergrund, kräftige Linienfarben, eine erkennbare Abstufung bei Farbtönen und Tonwerten sowie eine klare Freistellung zwischen den Gestaltungsmitteln eingehalten werden. Zusätzlich müssen optische Täuschungen vermieden oder möglichst gering gehalten werden. Ein größerer Kreis umgeben von kleineren wirkt beispielsweise häufig größer als wenn der gleiche Kreis von größeren Kreisen umgeben ist.

Zuletzt sei darauf hingewiesen, dass ein Kartennutzer auch Gewohnheiten bzw. Erwartungen an eine Karte stellt. Aus diesem Grund ist es zum Beispiel hilfreich, für Karten die gleiche oder ähnliche Inhalte widerspiegeln, einen analogen Kartengrund bzw. identische Gestaltungsmittel zu wählen.

6.2.3 Entwicklung der Softwarekarte IT-Masterplan

Verwendet man zur Beschreibung des IT-Masterplans und seiner Komponenten die Begrifflichkeiten des IEEE 1471 Standards, so ist der IT-Masterplan ein *Viewpoint* der Unternehmensarchitektur-Beschreibung. Um den Anforderungen unterschiedlicher Stakeholder gerecht zu werden und gleichzeitig eine Nutzen bringende Darstellung zu erzielen, die nicht aufgrund zahlreicher Informationen überladen ist, bietet sich die Aufteilung der Informationen in Form mehrerer Softwarekarten an. Im Fall

des IT-Masterplans handelt es sich um drei Karten, folgend als Ansicht bezeichnet, die mittels des gleichen *Viewpoints* beschrieben werden können.

Für den IT-Masterplan wurde der Kartentyp Intervallkarte gewählt, da die zeitlichen Abhängigkeiten der zu visualisierenden Projekte von entscheidender Bedeutung sind. Bei der Intervallkarte tritt der Aspekt *Zeit* in den Mittelpunkt, da sie als Dimension zur Verortung auf der x-Achse eingesetzt wird. Die Verortung auf der y-Achse erfolgt abhängig von der Ansicht in hierarchischer Form, wobei Objekte der gleichen Hierarchieebene alphabetisch anhand des Objektnamen sortiert werden. Ein schnelles Auffinden der gesuchten Objekte soll auf diese Weise ermöglicht werden.

6.2.3.1 Kartenzeichen

Die zu visualisierenden Informationsobjekte sind die Programme/Projekte der Fachbereiche, die Plattformprojekte und die Bausteine des Softwarearchitektur-Bebauungsplans. Gemäß dem Grundsatz *Gleiches gleich, Ungleiches ungleich* darzustellen, wurden für die Objekte unterschiedliche Gestaltungsmittel ausgewählt. Da die Intervallkarte als charakterisierende Notationstechnik Balken einsetzt, deren Länge die Dauer von Vorgängen, repräsentiert, kommen nur solche Gestaltungsmittel in Frage, die in ihrer Länge variable sind, ohne ihren Widererkennungswert zu verlieren. Wie Abbildung 48 zeigt, werden Plattformprojekte durch Rechtecke (Nr. 3), Fachbereichsprogramme/-projekte durch Sechsecke (Nr. 1) und Bausteine durch abgerundete Rechtecke (Nr. 6) mit jeweils unterschiedlichen Farben für die Füllung dargestellt.

Alle drei Gestaltungsmittel tragen den Namen des Objektes, um die eindeutige Identifikation der Objekte zu gewährleisten. Die Schrift ist sowohl horizontal als auch vertikal zentriert, die Schriftwahl fiel gemäß des Grundsatzes der Mindestgröße auf die serifenlose Schrift Arial der Größe 12pt fett. Die Kartenzeichen der Projekte werden horizontal entsprechend ihrer zeitlichen Ausdehnung ausgedehnt, die Höhe der Kartenzeichen wurde einheitlich für alle Zeichen auf 10 mm festgelegt.

Eine weitere Gestaltungsvariable der Kartenzeichen für die Programme, Projekte und Bausteine ist die Rahmenfarbe. Unterschiedliche Projektmeilensteine bzw. Bausteinaktivitäten werden jeweils durch das Kartenzeichen des Objektes mit entsprechender Kartenfarbe visualisiert. Da ein Programm nicht durch Meilensteine charakterisiert wird, ist dessen Rahmenfarbe stets schwarz. Für die anderen Klassen erfolgt die Wahl der Farbcodierung nach dem Ampelprinzip, um die Dringlichkeit des Handlungsbedarfes zu veranschaulichen. Befindet sich eine Plattform in der Analyse bzw. im Design, so trägt die Rahmenfarbe Rottönen (pink und rot), weil diese Plattform noch nicht in Projekten eingesetzt werden kann (vgl. Nr. 5). Bei den fachlichen Projekten ist die Realisierung bzw. Einführung in pink bzw. rot gefärbt, da hier unbedingt die benötigten Plattformen zur Verfügung stehen müssen (vgl. Nr. 2). Gibt es für einen Baustein bereits Standards, so wird dies mittels eines grünen Rahmens veranschaulicht, fehlt der Baustein vollständig, so ist die Rahmenfarbe rot (vgl. Nr. 7).

Für die Farben der Rahmen wurden kräftige und für die Füllung der Kartenzeichen möglichst unauffällige Farben gewählt, um dem Grundsatz des ausreichenden Kontrastes und der Objekttrennung gerecht zu werden.

Um den Zeitpunkt der Fertigstellung einer Plattformumgebung zu signalisieren, wird eine symbolische Signatur eingeführt, die entsprechend des Termins auf dem Plattformsymbol platziert wird. Es handelt sich dabei um einen Kreis, der in Anlehnung an den viertelstündigen Fortschritt eines Uhr Minutenzeigers, ausgefüllt wird. Ist das erste Viertel gefüllt, handelt es sich um das Symbol der Entwicklungsumgebung, ist die Hälfte gefüllt, um eine Integrationsumgebung. Die Qualitätssicherungsumgebung wird durch eine dreiviertel Füllung und die Produktion durch das vollständig eingefärbte Symbol visualisiert (Nr. 4). Die gewählte Symbolik erlaubt dem Kartenbetrachter den Rückschluss auf die Reihenfolge, in der die Umgebungen bereitgestellt werden. Es ist jedoch anzumerken, dass in einigen Fällen der HVB Systems, eine Plattform nicht immer alle Umgebungen bereitstellen muss. Diese Eigenschaft trifft insbesondere auf Plattformen zu, die Teil einer größeren Plattform sind.

Um einen Projektstop zu veranschaulichen wurde die Signatur eines Blitzes gewählt (Nr. 8). Der Zeitpunkt einer IT-Entscheidung wird mittels einer lila gefärbten Raute entsprechend des Datums auf dem Projekt oder Bausteinsymbol platziert (Nr. 9).

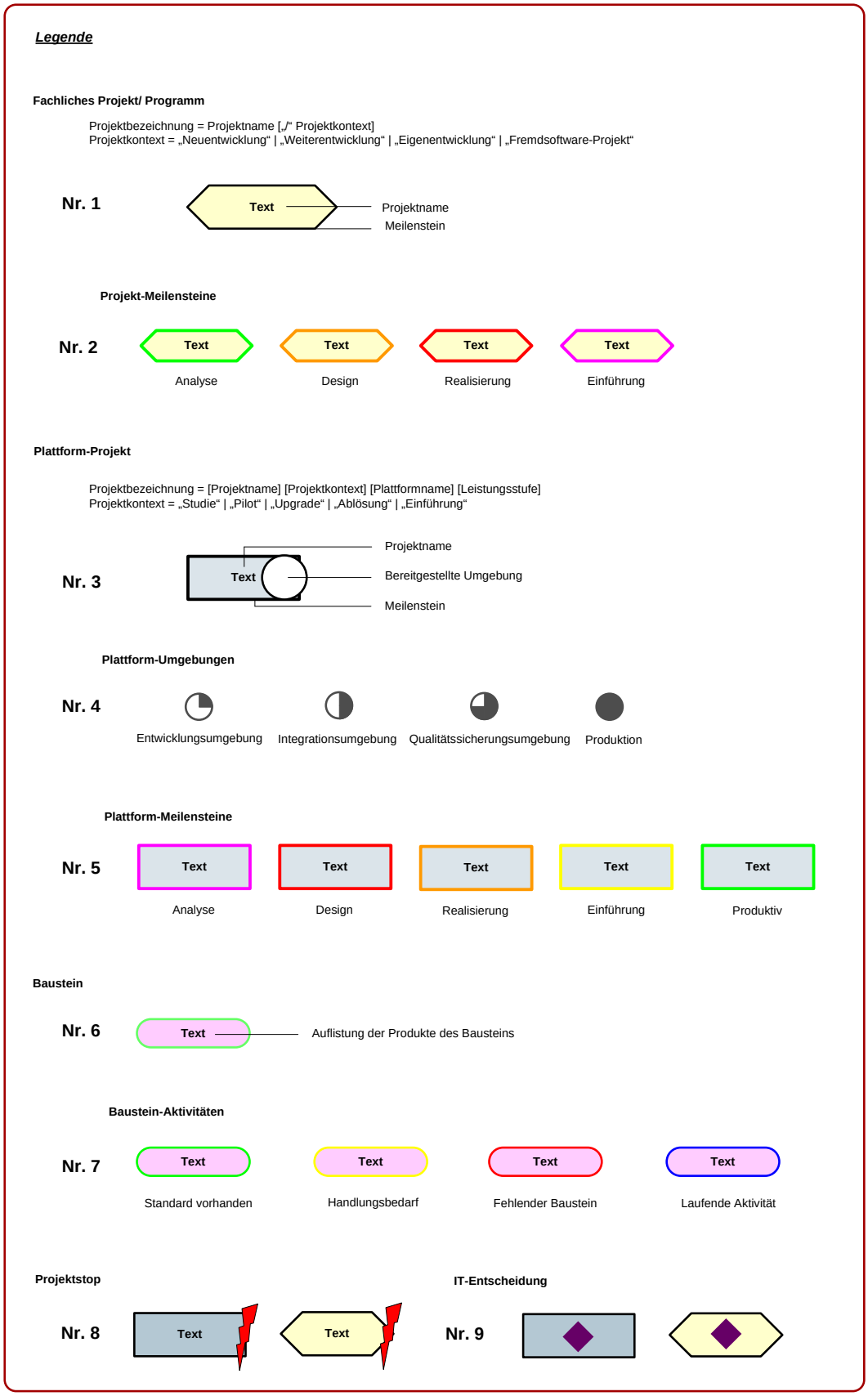


Abbildung 48: Legende des IT-Masterplans

6.2.3.2 Ansichten des IT-Masterplans

Nach dem die Kartenzeichen des IT-Masterplanes eingeführt wurden, soll nun auf die drei Ansichten Projekt, Plattform und Architektur eingegangen werden. Bei den beispielhaften Abbildungen handelt es sich dabei um anonymisierte Ausschnitte des für die HVB Systems entwickelten IT-Masterplans.

Die Anordnung von Elementen der drei Ansichten auf der y-Achse ähnelt der Ordnerstruktur des Microsoft Windows Explorer Fenster, die eine baumartige Verzweigung und Veranschaulichung der enthaltenen Objekte ermöglicht. Projektnamen können um einen Projektkontext ergänzt werden. Bei fachlichen Programmen/Projekten handelt es sich um die Ausprägungen „Neuentwicklung“, „Weiterentwicklung“, „Eigenentwicklung“ und „Fremdsoftware-Projekt“, bei den Plattformprojekten um „Studie“, „Pilot“, „Upgrade“, „Ablösung“ oder „Einführung“ (vgl. Abbildung 48 Nr. 1).

Mit Hilfe der *Projekt-Ansicht* soll die Terminierung von fachlichen Projekten und Plattformprojekten gesteuert werden können. Sie veranschaulicht die Zusammenhänge von Programmen bzw. Projekten der Fachbereiche und den von diesen Projekten eingesetzten Plattformen (siehe Abbildung 49). Die Struktur der y-Achse ist dementsprechend auf oberster Ebene an den Programmen ausgerichtet, für die auf zweiter Ebene beteiligte Projekte aufgelistet werden. In Abbildung 49 ist z.B. im Jahr 2005 nur das Projekt 2 im Rahmen des Programms 1 geplant, das jedoch im April gestoppt wurde. Da nicht jedes IT-Projekt einem Programm zugeordnet sein muss, kann die Angabe des Programms entfallen (vgl. fachliches Projekt 7 in Abbildung 49).

Für jedes Projekt werden auf der dritten Hierarchieebene Plattformen angegeben, die im Projekt eingesetzt werden bzw. eventuell eingesetzt werden können. Existieren für eine Plattform mehrere Versionen, so werden diese aufgeführt, um dem Entscheidungsträger mögliche Alternativen aufzuweisen. Dieser kann beispielsweise für das fachliche Projekt 2 in Abbildung 49 abwägen, ob er auf die bereits zur Verfügung stehende Plattform 3 in Version 1 aufsetzt oder besser wartet, bis Version 2 fertig gestellt ist um eine spätere Migration auf diese Version zu vermeiden. Für Projekt 7 kann der Entscheidungsträger anhand dieser Ansicht erkennen, dass es bei der „Realisierung“ zu Problemen kommen kann, wenn die benötigte Plattform 1 ihren Zeitplan nicht einhalten kann. Eine persönliche Abstimmung zwischen den Beteiligten des fachlichen Projekts 7 und dem Plattform-Projekt 1, könnte entsprechende Handlungsbedarfe aufdecken und einen reibungslosen Ablauf garantieren.

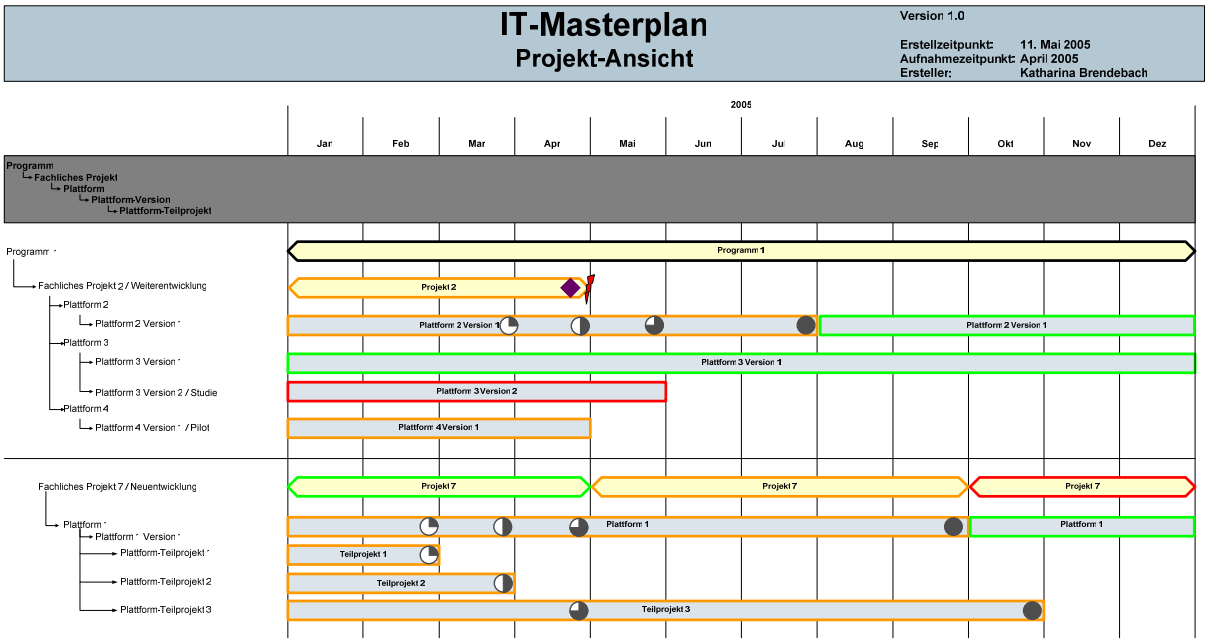


Abbildung 49: IT-Masterplan - Projekt-Ansicht

In der *Plattform-Ansicht* werden wie in der Projekt-Ansicht Informationen über die Zusammenhänge der Projekte aus den Fachbereichen und denen für die Plattformen dargestellt. In dieser Ansicht erfolgt jedoch die Ausrichtung der y-Achse auf oberster Ebene anhand der Plattformen, einschließlich deren Versionen (siehe Abbildung 50, Plattform 2 mit der einzigen Version 1). Ein Entscheidungsträ-

ger soll mit Hilfe dieser Ansicht in kurzer Zeit alle fachlichen Projekte auffinden können, die auf eine bestimmte Plattform bzw. Plattform-Version aufsetzten.

In der Praxis zeigte sich, dass Plattformen aus mehreren Plattform-Teilprojekten entstehen können, die getrennt voneinander durchgeführt werden und beispielsweise für die Fertigstellung der unterschiedlichen Plattform-Umgebungen sorgen. Aus diesem Grund kann auf der dritten Ebene eine Auflistung der Plattform-Teilprojekte vorgenommen werden (vgl. Plattform 1 Version 1 mit den Plattform-Teilprojekten 1-3 in Abbildung 50). Die Statusangabe der übergeordneten Plattform ergibt sich in diesen Fällen aus den Stati der Plattform-Teilprojekte, die von den Teilprojektverantwortlichen, an die Planer des Gesamtprojektes weitergeleitet werden müssen. Ein Entscheidungsträger kann beispielsweise in dieser Ansicht feststellen, dass bei der Planung der Plattform 1 Version 1 ein Fehler unterlaufen ist. Die „Produktion“ wird erst im Oktober 2005 durch das Teilprojekt 3 fertig gestellt, so dass die Plattform 1 Version 1 erst im November den Status „Produktiv“ erreicht. Eine Korrektur der Angaben kann vorgenommen und gegebenenfalls publiziert werden. Für das fachliche Projekt 1 hätte dies z.B. direkte Konsequenzen, da es sich im Oktober bereits in der „Realisierung“ befindet, und auf die Plattform 1 Version 1 aufsetzt. Es muss entschieden werden, ob entweder das Plattformprojekt früher fertig gestellt werden muss, oder das fachliche Projekt erst einen Monat später die Plattform verwendet.

Mit Hilfe der Plattform-Ansicht kann zudem eine Priorisierung der Plattform-Projekte vorgenommen werden. Besteht beispielsweise ein Ressourcenengpass, so dass die Realisierung einer Plattform eingefroren werden muss, kann diejenige Plattform ausgewählt werden, bei der die wenigsten fachlichen Projekte betroffen wären. Bei dem Szenario aus Abbildung 50 würde sicherlich die Plattform 2 Version 1 nicht abgebrochen werden, da sie im Gegensatz zu Plattform 1 bereits von fünf fachlichen Projekten eingesetzt werden soll.

Des Weiteren kann es für eine Plattform und ihre fachlichen Projekte entscheidend sein, ob sich die Version eines Produkts ändert. Diese Aktivitäten können auf der vierten Ebene der Plattform-Ansicht aufgeführt werden. Der Status der Plattform muss dabei nicht zwingend von den Änderungen einer Produktversion abhängig sein, z.B. bei kleinen Updates. Ein fachliches Projekt kann dahingegen entscheiden, ob es eine gewisse Zeit lang wartet, um direkt auf die neue Produktversion aufzusetzen. Bei einer neuen Version einer einzusetzenden Datenbank, könnte diese Abwägung z.B. Auswirkungen auf das Budget des fachlichen Projekts besitzen (vgl. Plattform 5 mit der Änderung des Produktes 1 auf die Version 3 und das darauf aufsetzende fachliche Projekt 8 in Abbildung 50).

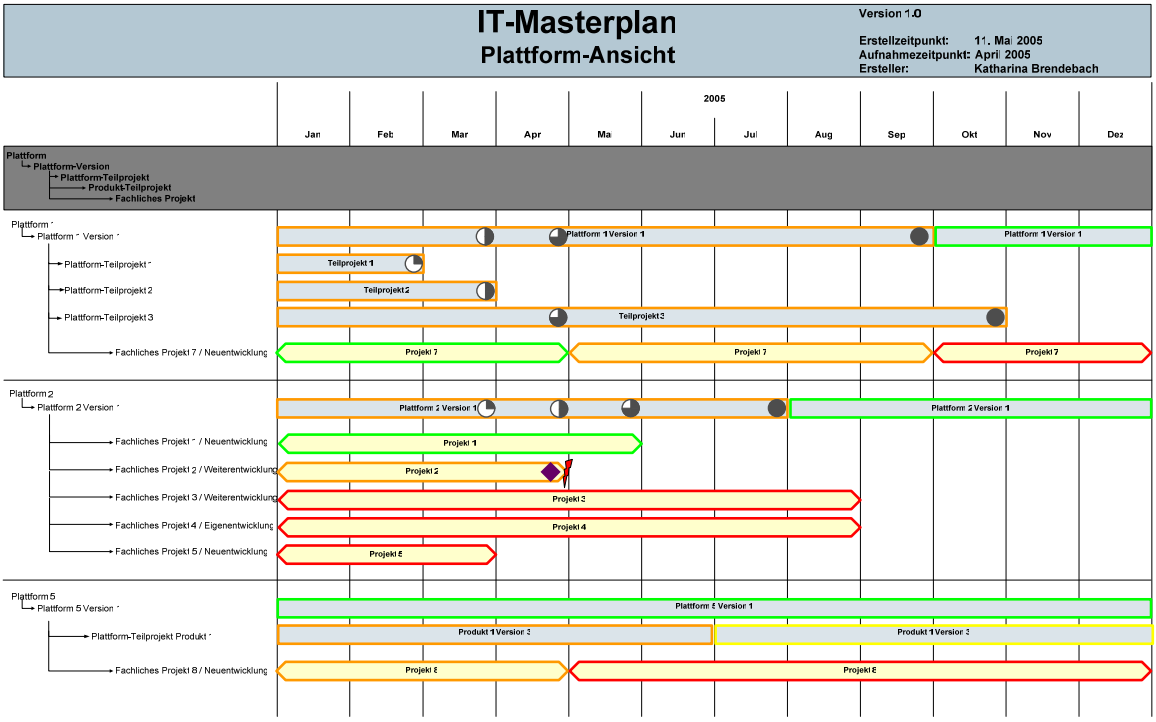


Abbildung 50: IT-Masterplan - Plattform-Ansicht

Die dritte Ansicht ist die *Architektur-Ansicht*, in der die Bestandteile einer Referenzarchitektur veranschaulicht werden können. In Abbildung 51 wird z.B. ein Ausschnitt des Softwarearchitektur-Bebauungsplans mit seinen Schichten und Bausteinen dargestellt. Der Status eines Bausteins ergibt sich dabei aus den ihm zugeordneten Produkten, die auf der vierten Hierarchieebene aufgelistet werden. Da beispielsweise dem Baustein 3 bislang keine Produkte zugeordnet wurden, besitzt er den Status „fehlender Baustein“.

Ändert sich eine Produktversion, so dass z.B. betroffenen Plattformen zu der neuen Version migrieren müssen, so wird diese Aktivität mittels der Symbolik für das Plattform-Projekt veranschaulicht (vgl. Baustein 1 mit den sich ändernden Produkten 1 und 2 in Abbildung 51).

Die Architektur-Ansicht dient somit dem Überblick über existierende bzw. fehlende Bausteine und Produkte.

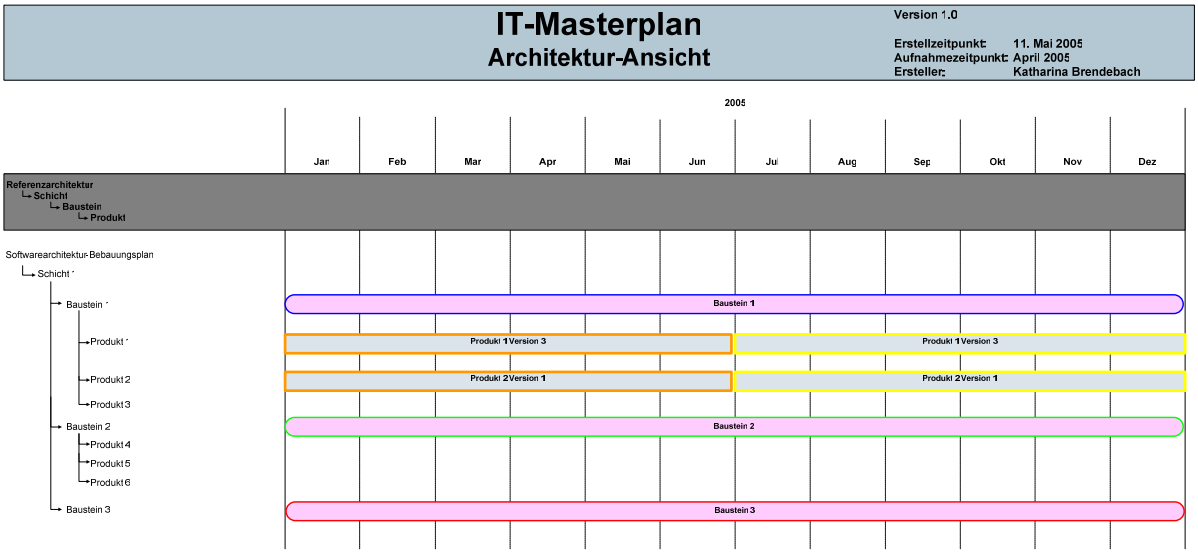


Abbildung 51: IT-Masterplan - Architektur-Ansicht

Der IT-Masterplan wurde in der vorliegenden Version mit Hilfe von Microsoft Visio erstellt. Es handelt sich demzufolge um ein statisches Dokument, das aus drei Zeichenblättern besteht und deren Inhalte händisch platziert wurden. Zukünftig soll die Erstellung des IT-Masterplans automatisiert ablaufen und den Stakeholdern mittels einer Anwendung zur Verfügung gestellt werden, die das Springen zwischen den Ansichten durch Doppelklick auf einzelne Objekte erlaubt. Der Stakeholder soll zunächst die Ansicht für den Einstieg in den IT-Masterplan wählen und kann dann, bei Interesse zu weiteren Informationen, zwischen den Ansichten wechseln. Zusätzlich soll mittels des Symbols für die IT-Entscheidungen ein Sprung ins Entscheidungsrepository ermöglicht werden, um beispielsweise den Grund für einen Projektstop herauszufinden.

Beginnt ein Stakeholder z.B. mit der Projekt-Ansicht, kann er sich einen Überblick über aktuell laufende fachliche Programme und Projekte einschließlich der Abhängigkeiten zu den Plattformen informieren. Interessiert er sich darüber hinaus für alle fachlichen Projekte, die auf einer gleichen Plattform aufsetzen, so kann er durch Doppelklick auf das Plattformsymbol zur Ansicht der Plattformen kommen. Entsprechend kann er über den Doppelklick auf ein fachliches Projekt zur ursprünglichen Ansicht zurück gelangen. Die Verknüpfung der Plattform und Architektur Ansicht kann über die technischen Produkte bzw. die sie betreffenden Projekte vorgenommen werden.

Aufgrund der drei Ansichten mit den unterschiedlichen Verortungskriterien auf der y-Achse wird den Stakeholdern ein Werkzeug bereitgestellt, das ihnen schnell gewünschte Informationen liefert.

6.2.3.3 Kartenpflege

Um den IT-Masterplan ständig auf dem aktuellen Stand zu halten, sollte zunächst der Abteilungsleiter bei der Initiierung eines Plattformprojektes die Aufnahme des Projektes in den IT-Masterplan veranlassen. Wurde dem Projekt ein Projektleiter zugeordnet, so ist dieser dafür verantwortlich, den aktuel-

len Status des Projektes einschließlich der restlichen Daten an den IT-Masterplan Verantwortlichen weiterzuleiten. Ist ein Plattformprojekt beendet und die Plattform als Standard vorhanden, so ist dies ebenfalls an den IT-Masterplan Verantwortlichen zu melden. Bei den Programmen und Projekten aus den Fachbereichen kann entsprechend vorgegangen werden, da spätestens nach der Analyse der Systemarchitektur ein Mitarbeiter der Architekturabteilung zugeordnet wurde, der somit für die Datenbeschaffung der für die IT-Masterplan relevanten Informationen verantwortlich gemacht werden kann. Die Aufgabe des IT-Masterplan Verantwortlichen ist darüber hinaus, bei neu hinzukommenden Informationen Anpassungen der Programme und Projekte sowie der Plattformprojekte vorzunehmen und die neuen Ergebnisse zu publizieren.

6.2.4 Abschließende Bemerkungen

Die erste Schwierigkeit lag in der Datenbeschaffung, die mittels unterschiedlichster Informationsquellen und *Information Supplier* händisch vollzogen werden musste. Auch die Qualität der Informationen für die erste Version ist nicht hundertprozentig, da beispielsweise Informationen über die Meilensteine von Projekten, sowie Angaben zu Plattformumgebungen unvollständig waren. Für die Visualisierung der Meilensteine wurde die jeweils letzte Phase, d.h. die Phase mit den maximalen Anforderungen an die Plattformen gewählt, um lieber zu früh als zu spät auf mögliche Abhängigkeiten hinzuweisen. Ebenso fehlten zu den Projekten genaue Angaben bezüglich der Entwicklungsumgebungen.

Die zweite Erkenntnis ist, dass die Gestaltung einer Softwarekarte einen immensen Aufwand in sich birgt. Die Wahl der Gestaltungsmittel und Variablen sowie die Farbwahl bedürfen einiger Entwurfsvarianten, bis eine erste akzeptable Version entsteht.

Kapitel 7

Zusammenfassung und Ausblick

Abschließend soll dieses Kapitel die Schwerpunkte und erbrachten Leistungen der Arbeit zusammenfassen, sowie einen Ausblick auf zukünftige Aufgaben für die Gestaltung der Unternehmensarchitektur unter Verwendung der Softwarekartographie geben (vgl. Abschnitt 7.1 und 7.2).

Das Kapitel 2 diente zunächst der Einbettung dieser Arbeit in das thematische Umfeld, indem die Begriffe IT-Management, Unternehmensarchitektur und Softwarekartographie erläutert, sowie miteinander in Bezug gesetzt wurden. In Kapitel 3 wurde daraufhin in die theoretischen Grundlagen eingeführt, die für das Verständnis der bearbeiteten Themengebiete benötigt wurden.

Die erste Teilaufgabe dieser Arbeit, d.h. die Analyse bestehender Diagramme und Karten der HVB Systems, wurde in Kapitel 4 vorgenommen. Sie bestätigte den steigenden Bedarf an unterschiedlichen Sichten der Unternehmensarchitektur, sowie deren Visualisierungen in Form von Softwarekarten. Identifizierte Sichten wurden aufgeführt, sowie die während der Analyse gewonnenen Erfahrungen zusammengefasst und als Anforderungen an das im Forschungsprojekt Softwarekartographie entstehende Werkzeug formuliert. Basierend auf dem Verständnis der Softwarekarten als eine Architekturbeschreibung und den Erkenntnissen der Analyse, wurde in Abschnitt 4.3 ein konzeptueller Entwurf des Visualisierungsmodells der Softwarekartographie entwickelt.

Die Entwicklung des integrierten Informationsmodells schloss sich an die Analyse an und stellte mit Kapitel 5 den zentralen Teil der Arbeit dar. Das Modell wurde in Kapitel 6 exemplarisch in Form von Sichten für die Anwendungslandschaft und der Softwarekarte IT-Masterplan, der die Abhängigkeiten und Querbeziehungen von fachlichen Projekten und Plattformprojekten in einem für Stakeholder geeignetem *Viewpoint* zusammenfasst, umgesetzt. Folgend werden die Hauptergebnisse dargestellt.

7.1 Ergebnisse

IT-Entscheidungsträger, insbesondere die des Finanzdienstleistungssektors, befinden sich häufig in einem Dilemma. Sie müssen fortlaufend schnell und effektiv auf eine stets wachsende Anzahl funktionaler Anforderungen reagieren und gleichzeitig Potential aufbauen, um in der Zukunft bestehen zu können. Agiert die IT auf rückwirkende Art und Weise, d.h. erfindet sie aktuell benötigten Produkte und Leistungen von Grund auf neu, anstelle durch erworbene Flexibilität auf Änderungen im Geschäft reagieren zu können, steigen die Betriebsgefahr und die Kosten der IT-Projekte [Kn02].

Um das Investment der IT besser auf die Geschäftstrategie und ihren Prozessen auszurichten, bedarf es einer optimalen Gestaltung der Unternehmensarchitektur [Kn04]. Um die entscheidenden Komponenten des Unternehmens und deren Zusammenspiel zu verstehen, sind high-level Architekturmodelle erforderlich (vgl. [Ja03]). In der vorliegenden Arbeit wurde für diesen Zweck ein Informationsmodell für das IT-Management entwickelt, dass 9 Pakete mit ca. 130 Entitäten sowie zahlreichen Assoziationen und Attributen umfasst (vgl. Kapitel 5). Um Anforderungen an das Modell zu identifizieren, wurden bestehende Diagramme und Karten der HVB Systems analysiert, die erste Hinweise auf die vom Management benötigten Sichten lieferten (siehe Kapitel 4). Zahlreiche Interviews sowie Diskussionen mit Mitarbeitern der HVB Systems halfen bei der Gestaltung der Teilmodelle und der Wahl eines geeigneten Abstraktionsgrades.

Das Informationsmodell stellt ein Werkzeug dar, das durch die in ihm vorgenommene Abstraktion der komplexen Sachverhalte eine hervorragende Basis für Diskussionen liefert. Teilmodelle können in Form spezieller Sichten implementiert werden, die dabei helfen, Handlungsbedarfe in der Unternehmensarchitektur zu identifizieren und Entscheidungen zu treffen (vgl. Kapitel 6). Abbildung 52 veranschaulicht die Komponenten des Informationsmodells (grün gefärbt) sowie die behandelten und entwickelten Sichten (gelb gefärbt).

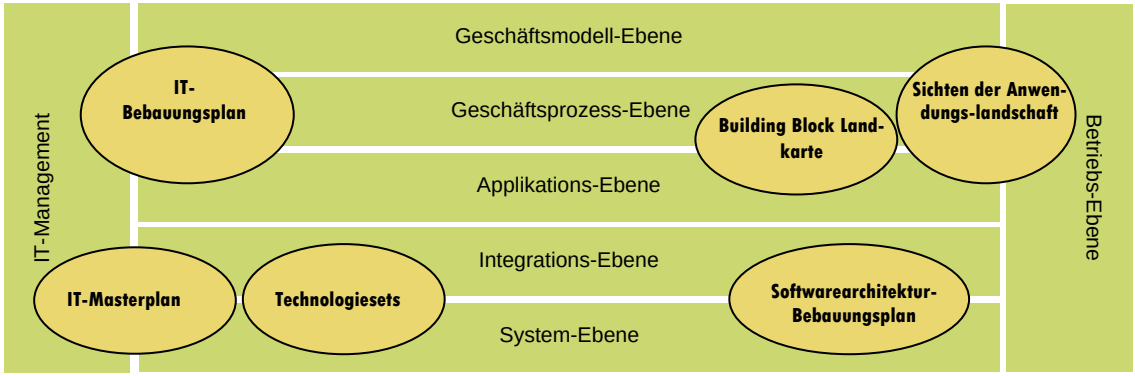


Abbildung 52: Übersicht der be- und erarbeiteten Elemente

Da die Unternehmensarchitektur für jedes Unternehmen individuell, unter Berücksichtigung der Geschäftstrategie und den Anforderungen an das Geschäft gestaltet werden muss [KK05], können sich Informationsmodelle verschiedener Unternehmen sowohl in ihrer Struktur als auch den relevanten Objekten unterscheiden. Zentrale Elemente, wie beispielsweise der Geschäftsprozess oder die Anwendungssysteme, werden jedoch in vielen Modellen aufzufinden sein. Im Folgenden soll dieser Tatbestand durch einen Strukturvergleich mit der von Lankes et al. [LMW05c] entwickelten Informationsmodell-Struktur (siehe Abbildung 53) veranschaulicht werden.

Das Modell von Lankes et al. stimmt mit dem der vorliegenden Arbeit teilweise überein. Die *Geschäftsebene* enthält ähnliche Klassen wie die Geschäftsmodell- und die Geschäftsprozess-Ebene, da sie Produkte, Geschäftsprozesse sowie Organisationseinheiten kapselt. Die *Applikationsebene*, welche die Geschäftsapplikationen einschließlich ihrer Komponenten und Schnittstellen beschreibt, kann mit der Anwendungssystem-Ebene gleichgesetzt werden. Auch für die in der *Infrastrukturebene* beschriebenen Infrastrukturkomponenten (z.B. Middleware- und Hardwaresysteme) lässt sich ein Gegenstück finden, da sie den Bestandteilen der Integrations- und Betriebs-Ebene ähneln (vgl. auch [Bo05]).

In einigen Punkten unterscheiden sich jedoch die beiden Modelle, da das Modell von Lankes et al. [LMW05c] modularen Charakter besitzt. Die Querschnittsfunktion *Kennzahlen und Metriken* kontrolliert und misst beispielsweise die Informationsobjekte der verschiedenen Ebenen und der anderen Querschnittsfunktionen. Dieser Gedanke spiegelt sich im Modell der vorliegenden Arbeit in den Attributen der Objekte wieder. Teile der drei anderen Querschnittsfunktionen *Ziele & Strategie*, *Projekte & Programme* und *Architekturen & Muster* sind in den Paketen der sechs Ebenen und dem IT-Management enthalten. Das Modell von Lankes et al. betont darüber hinaus sehr stark den serviceorientierten Gedanken, da es die beiden Ebenen *Geschäfts-Serviceebene* und *Infrastruktur-Serviceebene* beinhaltet, welche über Services die über- und untergeordneten Schichten verbindet, sowie die Applikationsebene kapselt. Das Pendant im Modell der vorliegenden Arbeit ist die Betriebs-Ebene mit der Definition von Vereinbarungen. Der Servicegedanke wird hier jedoch nicht so stark betont (s.u.).

Der strukturelle Vergleich der beiden Informationsmodelle für die Unternehmensarchitektur zeigt, dass die Modelle sehr ähnlich sind. Es ist nicht zwingend erforderlich ein Informationsmodell zu entwickeln, dass allgemein auf jedes Unternehmen übertragen werden kann. Vielmehr ist es entscheidend, dass sich ein Unternehmen über die Schlüsselkomponenten und Zusammenhänge der Unternehmensarchitektur im Klaren ist, um geeignete Sichten für einen optimalen Gestaltungsprozess der IT-Landschaft entwickeln zu können. Ein Informationsmodell, wie es in dieser Arbeit entwickelt wurde, kann dabei als Hilfestellung dienen.

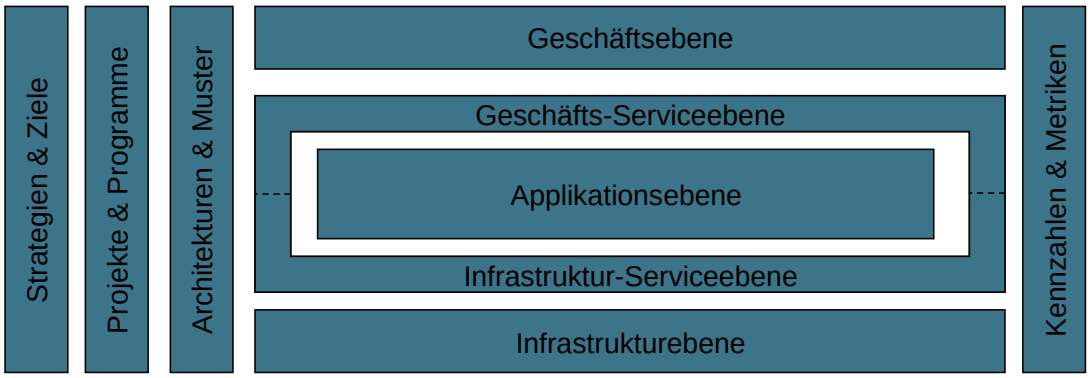


Abbildung 53: Unternehmensarchitektur Informationsmodell nach [LMW05c]

7.2 Ausblick

Eine erste nahe liegende Aufgabe liegt in der weiteren Umsetzung von Teilen des Informationsmodells. Den ersten Schritt stellt dabei die Erhebung der Daten für die ausgewählte Teilmenge an Klassen dar, die anschließend für die Erstellung weiterer Sichten genutzt werden können (vgl. Kapitel 6). Weitere Softwarekarten die die Zusammenhänge der Anwendungslandschaft bzw. Merkmale der Anwendungssysteme veranschaulichen, würden die Entscheidungsträger der IT in ihrer Arbeit unterstützen. Die Identifikation weiterer Sichten, sowie das Festlegen relevanter Kennzahlen können dabei als weitere Teilaufgaben angesehen werden.

Teile des Informationsmodells könnten darüber hinaus in einem nächsten Schritt verfeinert werden. Im Konkreten handelt es sich um das Paket des IT-Managements und der Betriebs-Ebene. Bei der Identifizierung relevanter Objekte für das IT-Management wurde deutlich, dass die Komponente des Architekturcontrollings näher betrachtet werden muss. Für eine Abteilung, die sich mit der Bereitstellung einer optimalen Architektur befasst, stellt sich die Frage, wie die Qualität der bestehenden Architektur geprüft werden kann. Regeln für die Bewertung und das Messen der Qualität sind dabei ebenso zu entwickeln, wie ein zugehöriger Prozess, der das gesamte Vorgehen von der Bewertung bis hin zu den die Architektur verbessernden Maßnahmen beschreibt. In Bezug auf das Paket der Betriebs-Ebene ist es vorstellbar, den Leistungs- bzw. Servicegedanken in die Gestaltung der Unternehmensarchitektur einzubinden, so dass er zu einem zentralen, die Architektur steuernden Teil wird. Dieser Ansatz wurde im zuvor eingeführten Modell von Lankes et al. [LMW05c] verfolgt. Es ist zu untersuchen, ob sich in der Praxis für die Komponenten der Geschäfts- und Infrastruktur-Serviceebene konkrete Ausprägungen erheben lassen, so dass diese zentrale Positionierung der Services berechtigt ist.

In der Arbeit wurde bereits häufig darauf hingewiesen, dass die Erstellung von Softwarekarten einen immensen Aufwand in sich birgt. Erste Konzepte und Grundlagen für eine automatische Kartengenerierung wurden durch Erläuterung des semantischen und symbolischen Modells einschließlich deren Komponenten dargelegt. Der bislang noch fehlende Prozess der Transformation zwischen den Modellen stellt jedoch ein offenes Gebiet dar.

Im Bereich der Informationsmodellierung selbst, liegt die Herausforderung in der Entwicklung einer Kosten/Nutzen Rechnung für eine umfassende Bewertung der Wirtschaftlichkeit eines Modells. Um effizient arbeiten zu können, soll ein Modellierer entscheiden können, ob anfallende Kosten für weitere Aktivitäten durch eine bessere Modellqualität gerechtfertigt werden oder nicht.

Ein weiteres interessantes Thema ist das Management von Lebenszyklen der Informationsobjekte, wie beispielsweise den Geschäftsprozessen, IT-Projekten, Anwendungssystemen oder dem Architekturmanagement-Prozess. Das Identifizieren und Beschreiben von bzw. der Umgang mit Objekten, die einen Lebenszyklus besitzen, einschließlich deren Abhängigkeiten, stellt einen interessanten Anknüpfungspunkt dieser Arbeit dar, um insbesondere den zeitlichen Wandel sowie den hieraus resultierenden Abhängigkeiten und Auswirkungen zwischen Objekten Rechnung tragen.

Die Ergebnisse der Arbeit sowie die Mannigfaltigkeit des Ausblicks zeigen, wie vielfältig und umfassend das Thema des Managements von Unternehmensarchitekturen ist.

Anhang A ANALYSE BESTEHENDER KARTEN

A.1. Building Block Landkarte

Die Building Block Landkarte (BB Landkarte) veranschaulicht für einen Building Block die zugehörigen Prozesse und die sie unterstützenden Anwendungssysteme. Zusätzlich werden Schnittstellen von Building Blocks, sowie die technische Bewertung von Anwendungssystemen visualisiert.

Einsatzgebiet

Zugänglich ist die BB Landkarte für alle Mitarbeiter über das Architekturportal PASS. Für jeden Building Block ist ein PDF Dokument hinterlegt, dass die mit Vision (Mikrosoft) erzeugte BB Landkarte abbildet. Kartennutzer ist somit die komplette PASS Zielgruppe. Hierzu zählen vor allem Projektmitarbeiter und Prozessspezialisten, IT-Spezialisten die relevante Informationen rund um Systeme, Plattformen und Standards benötigen, sowie alle Mitarbeiter, die sich einen Überblick über das Unternehmen schaffen wollen.

Auslöser für die Kartenerstellung

Die Entwicklung der BB Landkarte wurde durch die Zusammenarbeit mit der McKinsey Company in einem Projekt zur strategischen Unterstützung initialisiert.

Rolle	Rollenbelegung	Anzahl
Stakeholder	Projektmitarbeiter Prozessspezialist IT-Spezialist interessierter Mitarbeiter Architekt Projektmitarbeiter CIO	<100
Analyst	Architekt	<10
Modeler	Architekt	<10
Information Supplier	Architekt Anwendungssystem Verantwortlicher Building Block Leiter Vorstand Abstimmung mit Fachbereich	<100

Tabelle 1: Rollenbelegung der Building Block Landkarte

Rolle	Concerns und Questions
Stakeholder	**Concerns** Strukturierung der Anwendungslandschaft nach Prozess, Kunde und Produkt, Modularisierung, Flexibilität, Wirtschaftliche *Concerns* (Outsourcing, Wertschöpfungstiefe), Input / Output Analyse **Questions** Wie lassen sich die Dimensionen strukturieren und gruppieren? Wo lassen sich fachliche Schnittstellen definieren?
Analyst	**Questions** Welche Produkte werden mit welchen Prozessen unterstützt? Wie lassen sich Struktur und Transparenz der Anwendungslandschaft darstellen? Welchen Lebenszyklen-Status haben die Anwendungssysteme?

Tabelle 2: Concerns und Questions der Building Block Landkarte

Klassifikation nach Kartengrund

Den Rahmen der Building Block Landkarte bilden die 20 Building Blocks der HVB AG, die am äußeren Rand der Karte angeordnet sind. Innerhalb des BB Rahmens befindet sich ein Zoom-In eines der Building Blocks in Form eines grau gefüllten Rechtecks. Für diesen BB werden zunächst vertikal die einzelnen Teilprozesse visualisiert, wobei jeder Teilprozess die ihm zugeordneten Anwendungssysteme enthält.

Die Anordnung der Building Blocks am Kartenrand lässt zunächst vermuten, dass es sich um eine Clusterkarte handelt, bei der die Verortung anhand von logischen Einheiten vorgenommen wird, hier den Building Blocks. Da sie aber lediglich zur Veranschaulichung des Kartenfokusses und der Schnittstellenbeziehungen dienen und keine Rolle bezüglich der Verortung der Prozesse oder Anwendungssysteme spielen, muss es sich um einen anderen Kartentyp handeln. Bei genauerer Betrachtung lässt sich feststellen, dass eine definierte Verortung von Kartenelemente lediglich für die Anwendungssysteme, die sich innerhalb des Zoom Bereiches befinden, vorgenommen wird. Diese werden entsprechend der Teilprozesse eines Building Blocks platziert, so dass es sich bei der BB Landkarte um eine Prozessunterstützungskarte handelt. Die Verortung innerhalb der Teilprozesse wird beliebig vorgenommen. Auch die Building Blocks werden in beliebiger Reihenfolge platziert. Ein wichtiges Positionierungs- und Gestaltungskriterium stellt dabei die Größe eines DIN A4 Blattes dar, die einzuhalten ist, um ein einfaches Ausdrucken des Dokumentes zu gewährleisten.

Informationsobjekte

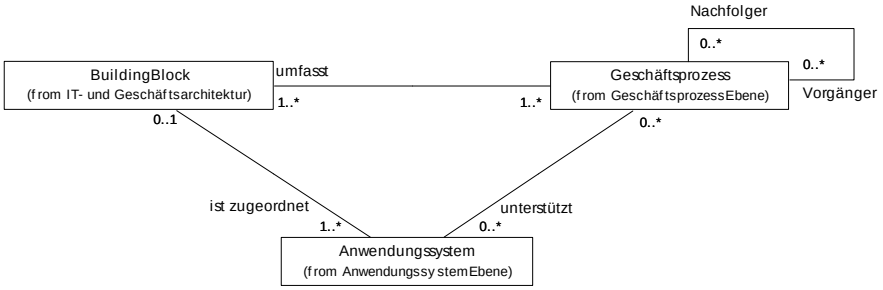


Abbildung 54: Informationsobjekte der Building Block Landkarte

Gestaltungsmittel

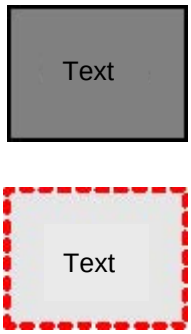
Element	Gestaltungsmittel	Notation	Beschreibung
Building Blocks	Rechteck mit Schrift		Beschriftung enthält Namen des BB und ist vertikal sowie horizontal „zentriert“ ausgerichtet. Der BB, auf den sich die Karte bezieht, besitzt eine dunkel graue Hintergrundfarbe und einen schwarzen Rahmen. Alle anderen Building Blocks sind hellgrau gefärbt und besitzen einen rot gestrichelten Rahmen.
Prozesse/ Teilprozesse	Rechteck mit Schrift		Die Beschriftung enthält Namen des Prozesses bzw. des Teilprozesse inkl. einer Prozessnummerierung. Ausrichtung ist vertikal "links" und horizontal "oben".
Anwendungssystem	Rechteck mit Schrift		Beschriftung enthält Namen des Anwendungssystems und wird vertikal und horizontal "zentriert" ausgerichtet. Eine vertikale Ausdehnung bedeutet, dass ein Anwendungssystem in mehreren Teilprozessen verwendet wird. Die horizontale Ausrichtung hat keine besondere Bedeutung sondern resultiert aus der Namenslänge.
Schnittstelle	Symbolische Signatur		Visualisiert Schnittstellenbeziehungen zu anderen BBs.
Technische Bewertung	Beschrifteter Kreis		Die Beschriftung T ist die Abkürzung für technische Bewertung. Sie wird vertikal und horizontal "zentriert" ausgerichtet. Bei einem roten Kreis handelt es sich um einen kurzfristigen Handlungsbedarf (0-18 Monate), bei gelber Färbung um einen mittelfristigen (18-36 Monate) und bei grün um einen langfristigen (36-60 Monate) Handlungsbedarf.
Funktionsabfolge	lineare Signatur		Verdeutlicht die Reihenfolge der Prozesse.

Tabelle 3: Gestaltungsmittel der Building Block Landkarte

Farbcodes (Schraffuren, Hintergründe, Farben für Rahmen)

Ursprünglich war für die Art der Schnittstellenbeziehung ein spezieller Farbcode vorgesehen, der jedoch derzeit keine Verwendung findet. Die Farbwahl für die technische Bewertung ist an das Ampelprinzip angelehnt, um die Dringlichkeit des Handlungsbedarfes zu signalisieren.

Schichtenprinzip

Es existieren keine Schichten, die ein oder ausgeblendet werden können, da die BB Landkarte im PDF Format vorliegt.

Selektive Informationsrepräsentation

Der Kartennutzer kann zwischen zwei unterschiedlichen PDF Dokumenten wählen: eine Ansicht der Building Blocks mit Prozessen und Anwendungssystemen und eine Ansicht mit zusätzlichen Angaben über die technische Bewertung der Anwendungssysteme.

Viewpoint

Im Sinne des IEEE Standards ist die Building Block Landkarte, ebenso wie der IT-Bebauungsplan, ein Modell zur Architekturbeschreibung der Anwendungslandschaft. Der zugehörige *Viewpoint* lässt sich als „Building Blocks der Anwendungslandschaft“ bezeichnen, da sie die Anwendungslandschaft hinsichtlich der Building Blocks, zugehörigen Prozessen und Anwendungssysteme filtriert. Der *Viewpoint* besteht dabei aus zwei Modellen (normale und technische Ansicht, siehe Abbildung 55 und Abbildung 56). Wie bereits beim IT-Bebauungsplan definiert, besteht auch bei der Building Block Landkarte der Kartengrund aus den Prozessen und Teilprozessen.

Initiale Kartenerstellung und anschließende Kartenpflege

Aktualisierungen dieser Karte werden nicht mehr vorgenommen, da die BB Landkarte durch eine aktuellere Darstellung in PASS ersetzt wird. Bei dieser PASS Version liegt der Redaktionsprozess in den Händen des PASS Release Managements. Es können alle 3 Monate größere Änderungen und wöchentlich kleinere Updates durchgeführt werden. Der Redaktionsprozess ist ein fest vorgeschriebener, aufwendiger und mit Budget Planung verbundener Ablauf, bei dem die Informationen von den Prozess- und Produktverantwortlichen verarbeitet werden.

Softwarekarten Beispiel

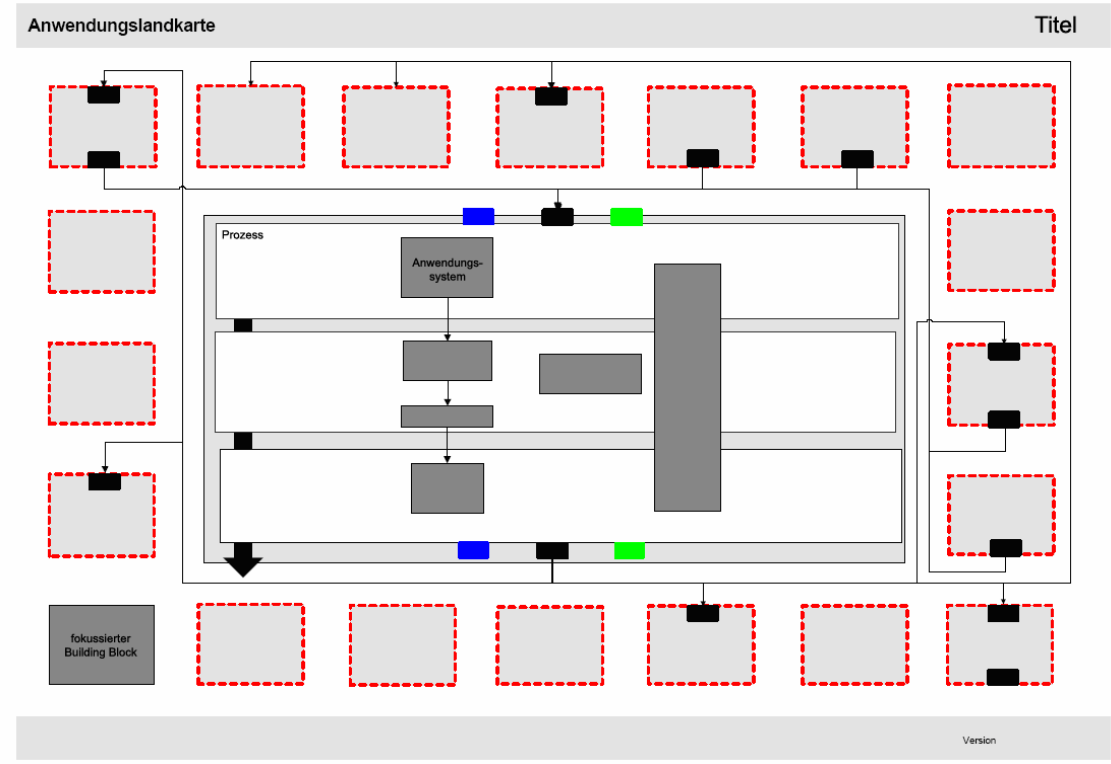


Abbildung 55: Building Block Landkarte

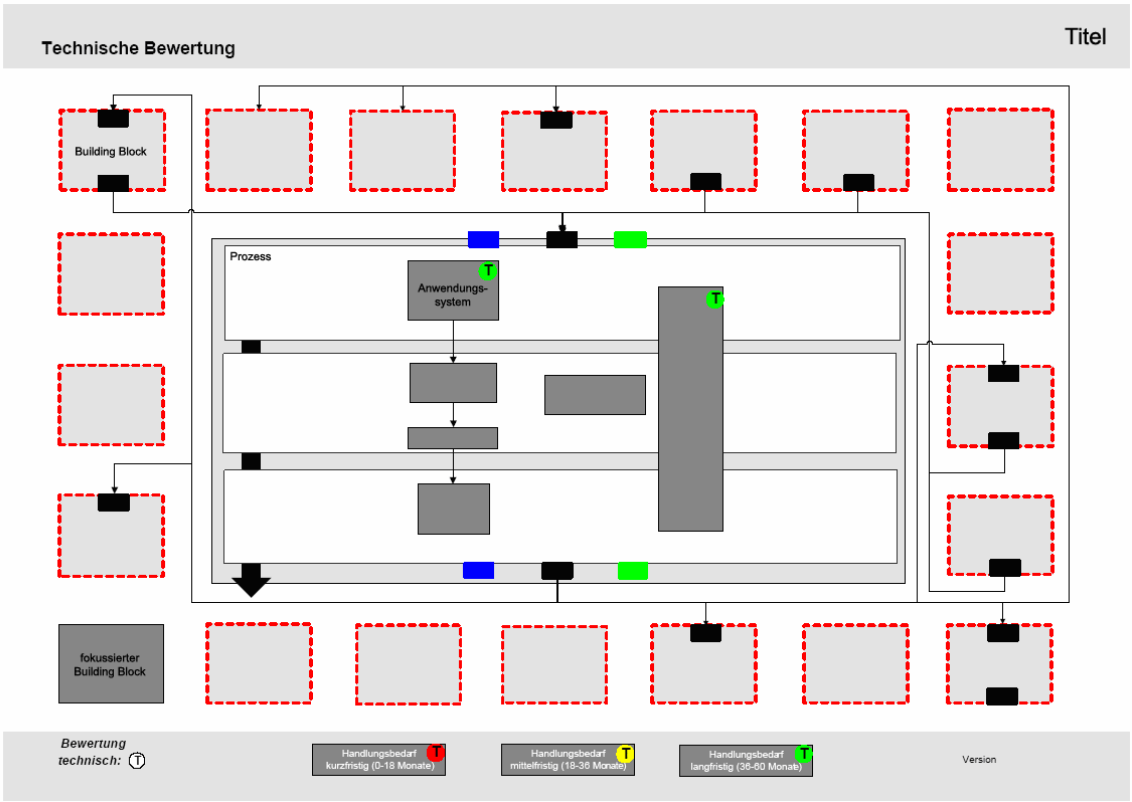


Abbildung 56: Building Block Landkarte - technische Ansicht

A.2. Softwarearchitektur-Bebauungsplan

Der Softwarearchitektur-Bebauungsplan (SA-BP) unterstützt die Umsetzung der IT-Strategie, da mit Hilfe seiner Komponenten, Anforderungen und Diensten definiert, sowie entsprechenden Objekten zugeordnet werden können. Er ermöglicht in Form eines Baukastens für Softwarearchitektur-Komponenten (Bausteine) die Klassifizierung von Produkten anhand ihrer Funktionalität, unterstützt die Produkt-Standardisierung und dient der Identifikation fehlender Bausteine bzw. Produkte.

Die wesentlichen Komponenten des SA-BPs sind die Bausteine. Sie stellen eine Gruppierung von Diensten dar und werden mit Hilfe von sieben Schichten kategorisiert. Jeder Baustein beinhaltete Attribute, die von der Anforderung bis zum Controlling reichen (z.B. Strategische Anforderungen, Key Performance Indikatoren, kurz KPI).

Durch das Rahmenwerk, welches der SA-BP mit seinen Bausteinen und Attributen definiert, lassen sich Produkte mittels ihrer Funktionalität bzgl. der relevanten Schichten bewerten und konkreten Bausteinen[37] zuordnen. Darauf aufbauend können optimale Zielplattformen definiert werden, welche die technologische Ausprägung eines Bausteins darstellen. Plattformen bestehen aus ein oder mehreren Technologiestandards und bündeln Dienste entsprechend der Bausteine (z.B. J2EE, JMS, JNDI). In einem weiteren Schritt können jeweils Bausteine der Schichten des SA-BP zu Technologieset zusammengefasst werden.

Folglich bildet der SA-BP die Möglichkeit, die Standardisierung mangelhafter Produkte zu verhindern, in dem nur solche Produkte in das Sortiment aufgenommen werden, die auch entsprechend der Schichten geeignete Funktionalitäten liefern. Vor allem können fehlende Bausteine und somit auch fehlende Produkte ermittelt werden, in dem überprüft wird, ob alle Anforderungen der Schichten durch Bausteine erfüllt werden.

Die wichtigste Eigenschaft des SA-BP ist jedoch, dass mit seiner Hilfe die IT-Strategien über Anforderungen in den Bausteinen und somit in den konkreten Produkten zutragen kommen.

Einsatzgebiet

Der SA-BP ist ein Werkzeug für den Architekten. Er kommt nicht direkt in den Projekten zum Einsatz, sondern dient vielmehr im Bereich des Architekturmanagement-Prozesses der Definition von Bausteinen und ihren Attributen, sowie der Bewertung von Produkten. Der besondere Fokus liegt auf dem Erkennen des Bedarfes an neuen Bausteinen und somit auch neuen Produkten, die den Anforderungen der Schichten gerecht werden.

Genutzt wird die Karte von Architekten und Architektur Beteiligten der HVB Systems, besonders im Rahmen von Gremien die sich mit Architekturentscheidungen befassen. Zukünftig soll der SA-BP auch im Architekturportal PASS integriert werden, so dass jeder Mitarbeiter über das Intranet Zugriff auf den SA-BP hat. In welchem Maß dieses Angebot nur als Informationszweck dient oder der SA-BP konkret genutzt wird, ist abzuwarten.

Umgeben ist der SA-BP von den IT-Referenzarchitekturen und den Technologiesets. Die IT-Referenzarchitekturen beschreiben Lösungsbausteine bzw. Patterns, aus denen dann projektspezifische Architekturen abzuleiten sind. Es werden Angaben gemacht, welche Rollen die Bausteine einnehmen und über welche Schnittstellen sie miteinander kommunizieren. Die Aufgabe der Technologiesets liegt insbesondere in der Definition einer gültigen und sinnvollen Kombination von Produkten, sowie dessen Lebenszyklus-Management. Sie dienen der Kommunikation in Projekten.

Auslöser für die Kartenerstellung

Die Idee für einen Bebauungsplan wurde durch den jetzigen Leiter der Architekturabteilung eingeführt. Ausgangspunkt ist die Verwirklichung der IT-Strategie, die konkrete Auswirkungen auf die IT-Architektur ausübt.

Die Definition der sieben Schichten, die jeweils unabhängig von einander sind, dient einer Abstraktion der IT-Architektur. Strategische Anforderungen sollen Schritt für Schritt in konkreten Bausteinen abgebildet werden. Das bedeutet, dass die Bausteine vor allem der Kapselung von Anforderungen,

[37] gegebenenfalls auch mehreren Bausteinen gleichzeitig

Standardisierung und Qualitätssicherung dienen. Entscheidend ist, dass jeder Baustein unabhängig von anderen Bausteinen geprüft, eingesetzt und vor allem gesteuert werden kann. Auch der Lebenszyklus einzelner Bausteine mit seiner konkreten Realisierung ist eine entscheidende Einflussgröße, die durchwegs lenkbar sein muss.

Rolle	Rollenbelegung	Anzahl
Stakeholder	Projektarchitekt Architekt mit einer strategischen Sicht auf die Bausteine und Produkte Auftraggeber (z.B. Fachbereich, Konzern-Architekt)	<100
Analyst	Architekt	<10
Modeler	Architekt	<10
Information Supplier	Produktverantwortlicher Schichtverantwortlicher	<1000

Tabelle 4: Rollenbelegung des Softwarearchitektur-Bebauungsplans

Rolle	Concerns und Questions
Stakeholder	**Concerns** Die IT-Strategie soll in den Projekten zu tragen kommen. Einsatz eines IT-Governance Werkzeuges **Questions** Projektsicht: Wie kann gewährleistet werden, dass spezielle Anforderungen von Produkten erfüllt werden? Wie kann der Standardisierungsgedanke umgesetzt werden? Steigender ROI: Wie kann langfristige Kostenreduktion erzielt werden? Werden KPIs durch einen Bausteinen eingehalten? D.h. rentiert sich der Einsatz eines Produktes?
Analyst	**Questions** Wie werden Anforderungen über Bausteine und dessen Dienste transparent? Wie kann die Standardisierung mangelhafter Produkte verhindert werden? Werden alle strategischen Anforderungen durch Schichten repräsentiert? Werden alle Anforderungen der Schichten durch Bausteine und dessen Dienste zur Verfügung gestellt? Sind alle Bausteine durch Produkte realisiert? Liefert ein Produkt entsprechend der Schichten geeignete Funktionalität? Welche Schichten, Bausteine, funktionalen Anforderungen und Attribute gibt es? Wie ist die Struktur des Plans?

Tabelle 5: Concerns und Questions des Softwarearchitektur-Bebauungsplans

Klassifikation nach Kartengrund

Die Verortung der Kartenelemente (Bausteine und Produkte) wird nicht entsprechend der x- oder y-Achse vorgenommen, sondern die Elemente werden in logischen Einheiten d.h. Clustern geordnet. Bei der vorliegenden Karte bilden die sieben Schichten die Cluster, es handelt sich demzufolge um eine Clusterkarte.

Die relative Position ist bei Bausteinen und Produkten relevant, die aufgrund ihrer gemeinsamen Zugehörigkeit zu Schichten, möglichst nah beisammen liegen.

Informationsobjekte

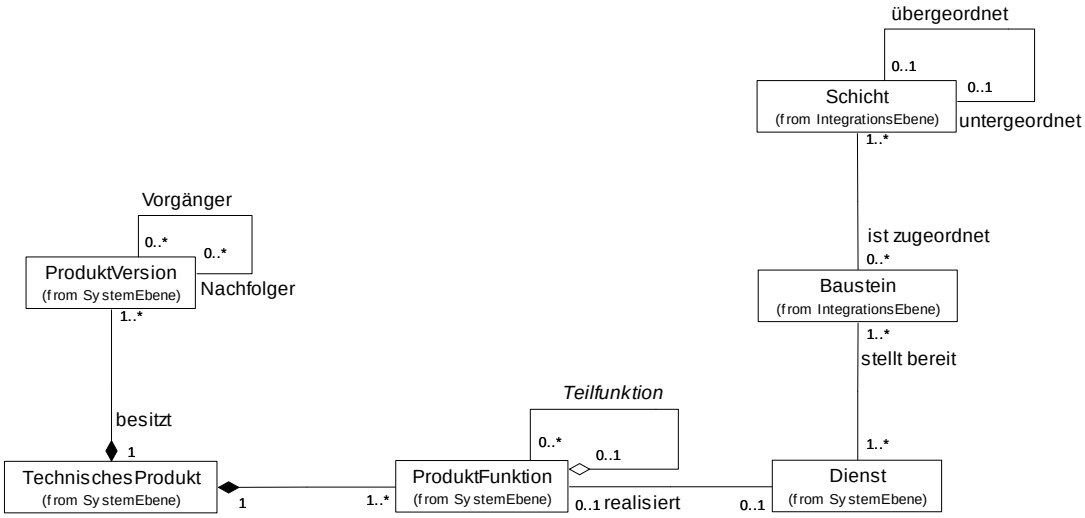


Abbildung 57: Informationsobjekte des Softwarearchitektur-Bebauungsplans

Gestaltungsmittel

Der SA-BP befindet sich derzeit noch in der Entwicklungsphase. Der Ausgangspunkt für eine graphische Repräsentation war zunächst eine Excel-Datei (Microsoft), aus der in einem ersten Schritt eine Darstellung in Form von Powerpoint Graphiken entstand. Hierauf aufbauend wurde zum Zeitpunkt dieser Arbeit eine zweite Version mit Hilfe des ARIS Toolsets (IDS Scheer) realisiert, welche folgend genauer analysiert wird.

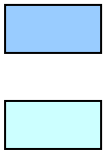
Element	Gestaltungsmittel	Notation	Beschreibung
Schicht	Symbolische Signatur		Die Beschriftung enthält den Namen der Schicht. Sie wird vertikal und horizontal „zentriert“ ausgerichtet.
Baustein	Symbolische Signatur	 Text	Die Beschriftung enthält den Namen des Bausteins. Sie wird vertikal unterhalb der Symbolik und horizontal „zentriert“ platziert.
Schicht-Zugehörigkeit	Rechteck		Um die Zugehörigkeit der Bausteine zu den Schichten unmissverständlich zu visualisieren, werden jeweils die Schicht und dessen Bausteine innerhalb eines blauen Rechteckes platziert. Die Querschnittsschicht erhält wegen ihrer Sonderstellung einen helltürkis farbigen Hintergrund.
Produkt	Symbolische Signatur	 Text	Die Beschriftung enthält den Namen des Produktes. Sie wird vertikal unterhalb der Symbolik und horizontal „zentriert“ platziert.
Produkt-Status	Rechteck mit Schrift		Sowohl Farbe als auch Beschriftung geben den Status von Produkten an. Die Beschriftung wird vertikal und horizontal „zentriert“ ausgerichtet.

Tabelle 6: Gestaltungsmittel des Softwarearchitektur-Bebauungsplans

Farbcodes (Schraffuren, Hintergründe, Farben für Rahmen)

In zwei Fällen tragen die unterschiedlichen Farben von gleichen Symbolen eine Bedeutung. Zum einen wird durch unterschiedliche Farben eine deutliche Abgrenzung zwischen der Querschnittsschicht und den restlichen Schichten bewirkt. Zum anderen verweist die Farbwahl der Produkt-Status Rechtecke aufgrund der drei Ampelfarben grün, gelb und rot auf die Dringlichkeit des Handlungsbedarfes.

Schichtenprinzip

Es existieren keine Schichten, die ein oder ausgeblendet werden können.

Selektive Informationsrepräsentation

Der Kartennutzer kann zwischen zwei Ansichten wählen. Startpunkt des SA-BP ist eine Ansicht der Schichten mit den zugeordneten Bausteinen (siehe Abbildung 58). Durch Klicks auf einzelne Bausteine kann eine zweite Sicht auf die Standards mit den Produkten geöffnet werden (siehe Abbildung 59).

Viewpoint

Im Sinne des IEEE Standards kann für den Software Architektur Bebauungsplan der *Viewpoint* „Softwarearchitektur“ definiert werden, der sich aus zwei Modellen (Softwarearchitektur-Bebauungsplan und Bausteinansicht) zusammensetzt.

Initiale Kartenerstellung und anschließende Kartenpflege

Aktualisierungen des SW-BP werden durch Änderungen der Produktpalette und neuen Anforderungen initiiert. Solche Veränderungen müssen zunächst in einem Entscheidungsgremium abgestimmt und abgesegnet werden. Das zuständige Gremium findet in Abständen von zwei Monaten statt, sodass maximal in diesen Abständen Änderungen im SW-BP anstehen könnten.

Da sich der SA-BP derzeit noch in der Entwicklung befindet, sind keine genauen Angaben über die Kartenpflege möglich. Wird der SA-BP mit Hilfe des ARIS Toolsets (IDS Scheer) realisiert und in PASS veröffentlich, so ist noch eine genaue Methodik für die Initiale Datenfüllung und die fortlaufende Datenpflege zu entwickeln. Die Report Funktion des Werkzeuges könnte dabei eine entscheidende Rolle übernehmen.

Softwarekarten Beispiel

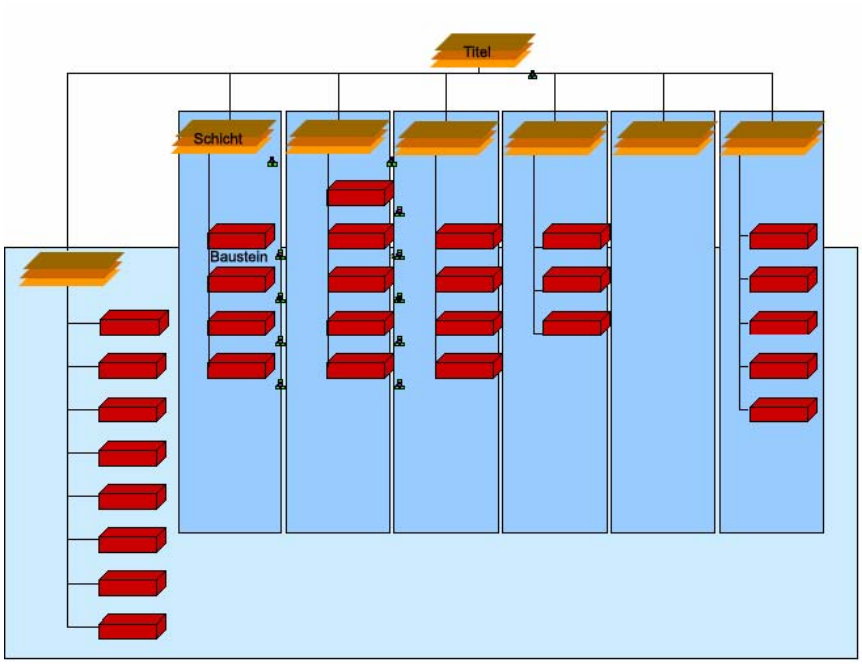


Abbildung 58: Softwarearchitektur-Bebauungsplan

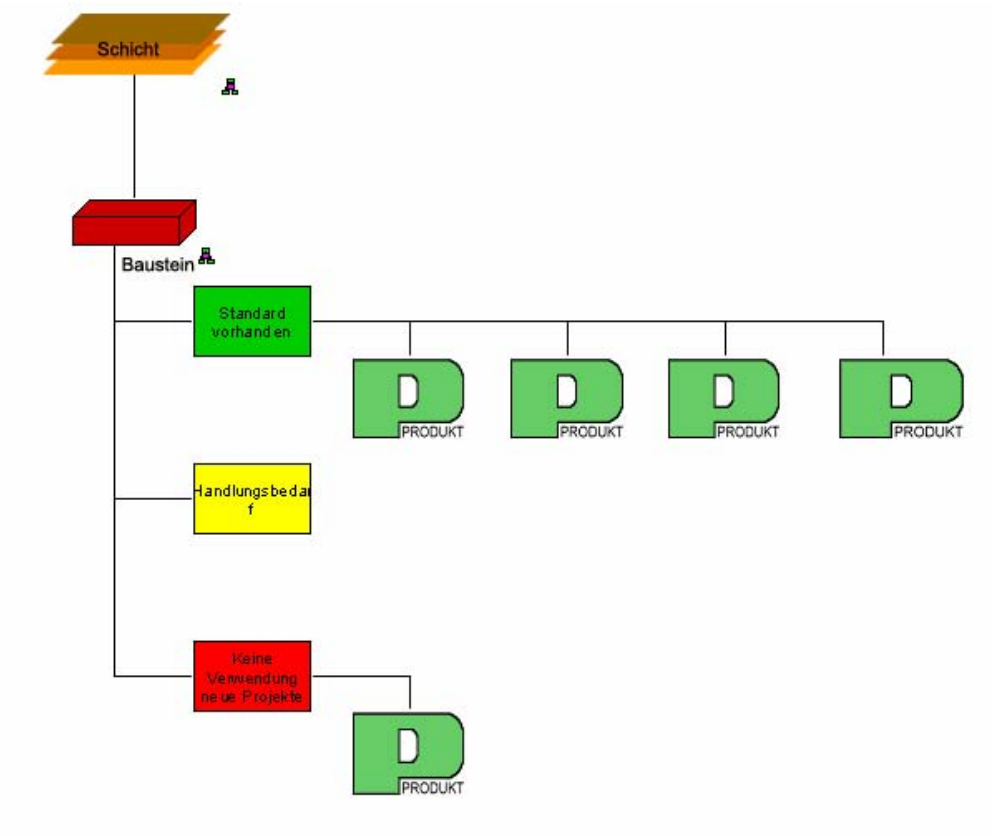


Abbildung 59: Softwarearchitektur-Bebauungsplan – Bausteinansicht

A.3. Technologieset-Karte

Die Technologieset-Karten liefern einen Überblick über Produkte, Produktkombinationen (Technologiesets) und Produktversionen, die in Projekten eingesetzt werden können. Sie helfen bei dem Management des Produktlebenszyklus und unterstützen die Architekturstandardisierung.

Technologiesets stellen die wichtigste Komponente dieser Karte dar. Sie beschreiben eine für die Entwicklung einsetzbare gültige und funktionierende Kombination von Produkten, inklusive Versionsnummern und ihren aktuellen Lebenszyklus-Status. Dieser Lebenszyklus wird ebenfalls in den Technologieset-Karten dargestellt und ermöglicht es, den Gültigkeitszeitraum von Produktkombinationen zu managen und somit frühzeitig auf Veränderungen zu reagieren. Ziel der Technologiesets und deren Visualisierung besteht darin, eine ganzheitliche Sicht auf die in einem bestimmten Anwendungsgebiet eingesetzten Technologien zu erhalten. Es soll für einen Anwendungsentwickler möglich sein, anhand von den gegebenen Projektanforderungen das geeignete Technologieset auszuwählen.

Einsatzgebiet

Da die Technologiesets im Architekturportal PASS veröffentlicht werden, umfasst die Kartennutzer die komplette PASS Zielgruppe. Besonderen Einsatz finden die Technologiesets und ihre Karten vor allem in Projekten, in denen die Projektmitarbeiter mit Hilfe der Technologiesets entscheiden können, welche Produkte in welcher Kombination im Projekt eingesetzt werden können.

Auslöser für die Kartenerstellung

Initiierung der Technologieset Entwicklung und deren graphischer Repräsentation bestand in der Notwendigkeit, die Vielfalt der in der Firma verwendeten Produkte und Produktkombinationen zu reduzieren. Betreiberkosten sollten minimiert werden und Umgebungen z.B. für die Qualitätssicherung oder anderen Sicherheitsmaßnahmen zuschaffen.

Der Vorgang zur Definition der Technologiesets war eine ausgiebige Bestandsaufnahme. Es wurde ermittelt, welche Produkte und Produktkombinationen im Einsatz waren um anschließend geeignete Kombinationen zu Technologiesets zusammenzufassen.

Grundlage für die Gestaltung der Technologiesets Karte bilden Produktsteckbriefe in Form von Excel-Dokumenten (Microsoft), die alle wesentlichen Informationen zum Produkt und dessen Bewertung durch die Produktverantwortlichen beinhalten.

Die Technologieset-Karte liefert somit eine Visualisierung der Dokumentinhalte, um die textuelle Repräsentation der Daten durch graphische Mittel gegenüber Dritten zugänglicher zu machen.

Rolle	**Rollenbelegung**	**Anzahl**
Stakeholder	Architekt Projektleiter PASS Zielgruppe	>1000
Analyst	Architekt	<10
Modeler	Architekt	<10
Information Supplier	Technologieset Owner (pflegt, aktualisiert und veröffentlicht die Gesamtheit der Technologiesets) Pate (ist für eine Gruppe von Technologiesets verantwortlich, pflegt und aktualisiert einen Teil der Produktsteckbriefe; direkter Ansprechpartner für den Produktverantwortlichen) Produktverantwortlicher (ist für alles verantwortlich, was mit einem Produkt zusammenhängt, z.B. Planung, Betrieb, etc.) Technologieset-Komitee (tritt vierteljährlich zusammen und beschließt den Einsatz bzw. die Verfahrenweise mit neuen und alten Technologiesets.) Hilfsmittel der Datenbeschaffung ist das Firmenrepository	<1000

Tabelle 7: Rollenbelegung der Technologieset-Karte

Rolle	Concerns und Questions
Stakeholder	**Concerns** Werkzeug für die Architekturstandardisierung (Kostenreduzierung, technische Homogenität) Richtlinien für den Einsatz von Produkten **Questions** Wie kann Architekturstandardisierung unterstützt und umgesetzt werden? Wie kann der Einsatz von Produkten geeignet gesteuert werden? Wie lässt sich der Produkt Lebenszyklus managen?
Analyst	**Questions** Welche standardisierten Produkte gibt es? Wo findet ein Produkt Verwendung? In welchem Technologieset? Welche Versionen existieren zu einem Produkt? Welche Produktkombinationen sind gültig d.h. welche Technologiesets existieren? Welches Technologieset ist in einem gegebenen Projektumfeld einzusetzen? In welchem Lebenszyklus befinden sich die Produkte und ihre Versionen (abgeschafft, wird ersetzt)? Welchen Handlungsbedarf hat eine Produktversion (kein Handlungsbedarf, Handlungsbedarf, kritisch, soll für neue Projekte)?

Tabelle 8: Concerns und Questions der Technologieset-Karte

Klassifikation nach Kartengrund

Bei der Technologieset-Karte handelt es sich um eine Matrixkarte. Diese ordnet ihre Kartenelemente in Tabellenform an, wobei die Verortung in Spalten und Zeilen nach den auf der x- und y-Achsen definierten Eigenschaften vorgenommen wird.

Bei der Technologieset-Karte gibt die x-Achse Auskunft über den Handlungsbedarf der Produktversionen, die y-Achse legt Einsatzort und Schicht der Produkte fest, um den Zusammenhang mit der Integrations- bzw. System-Ebene zu verdeutlichen.

Informationsobjekte

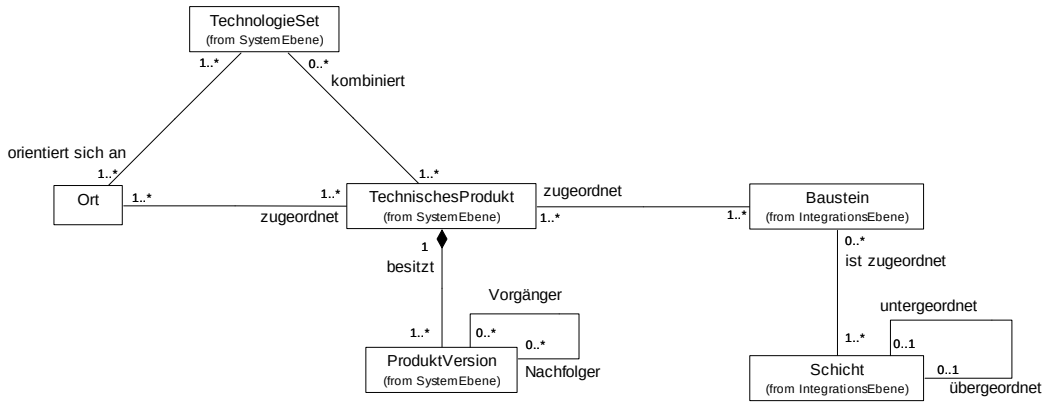


Abbildung 60: Informationsobjekte der Technologieset-Karte

Gestaltungsmittel

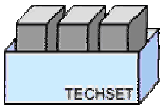
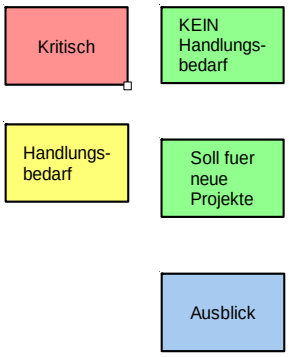
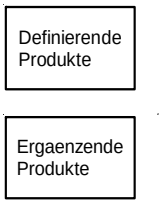
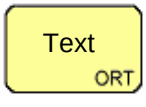
Element	Gestaltungsmittel	Notation	Beschreibung
Technologieset	Symbolische Signatur	 Text	Beschriftung enthält Namen des Technologiesets und wird vertikal unterhalb der Symbolik sowie horizontal „zentriert“ platziert.
Technisches Produkt	Symbolische Signatur	 Text	Die Beschriftung enthält den Namen des Produktes. Sie wird vertikal unterhalb der Symbolik und horizontal „zentriert“ platziert.
Produktversion	Symbolische Signatur	 Text	Die Beschriftung enthält den Namen der Produktversion. Sie wird vertikal unterhalb der Symbolik und horizontal „zentriert“ platziert.
Lebenszyklus-Status	Rechteck mit Schrift		Sowohl Farbe als auch Beschriftung geben den Status des Lebenszyklus von Produktversionen an. Die Beschriftung wird vertikal und horizontal „zentriert“ ausgerichtet.
Klassifikation von Produkten	Rechteck mit Schrift		Dient der Bewertung von Produkten, wird in zukünftigen Versionen nicht mehr verwendet. Die Beschriftung wird vertikal und horizontal „zentriert“ ausgerichtet.
Schicht	abgerundetes Rechteck mit Schrift		Die Beschriftung enthält den Namen des Ortes. Sie wird vertikal und horizontal "zentriert" platziert.
Ort[38]	Symbolische Signatur mit Schrift		Die Beschriftung enthält den Namen des Ortes. Sie wird vertikal und horizontal „zentriert“ platziert.

Tabelle 9: Gestaltungsmittel der Technologieset-Karte

[38] Der Ort ist im Informationsmodell nicht enthalten, da er aus Sicht der Unternehmensarchitektur nicht relevant ist.

Ergänzung:

Ausdehnungen von Objekten spielen keine Rolle. Sollte ein Produkt in mehreren Schichten oder Orten zu liegen kommen, so wird es erneut mit seinem Symbol hinterlegt.

Farbcodes (Schraffuren, Hintergründe, Farben für Rahmen)

Der Produktversions-Status kann als eine Kennzahl mit definierten Toleranzwerten interpretiert werden [Be04]. Um das Über- bzw. Unterschreiten des Toleranzwertes zu visualisieren, wurden zunächst unterschiedliche Warnstufen („Kritisch", „Handlungsbedarf", „kein Handlungsbedarf", „soll für neue Projekte" und „Ausblick") definiert und mittels der Symbolfarbe gemäß der Ampeldarstellung visualisiert.

Schichtenprinzip

Es existieren keine Schichten, die ein oder ausgeblendet werden können.

Selektive Informationsrepräsentation

Neben der eigentlichen Darstellung der Technologiesets (siehe Abbildung 61) existieren zwei weitere Ansichten. Die erste Ansicht dient der Übersicht aller Technologiesets für ein bestimmtes Einsatzgebiet (vgl. Abbildung 62), die zweite gibt dem Kartennutzer Informationen über die Verwendung eines Produktes in den Technologiesets (siehe Abbildung 63).

Viewpoint

Im Sinne des IEEE Standards kann für die Visualisierung der Technologiesets der *Viewpoint* „Technologieset" definiert werden, der sich aus drei Modellen zusammensetzt.[39]

Initiale Kartenerstellung und anschließende Kartenpflege

Die Kartenaktualisierung geht einher mit der Veröffentlichung neuer Releases. Aktualisierungen der Technologiesets werden durch Änderungen der Produktpalette inkl. Versionen initiiert. Änderungen sind im Bereich von 3 Monaten zu erwarten und können mit Hilfe der Report Funktion von ARIS Toolset (IDS Scheer) in gewissem Grad automatisiert werden.

[39] Da es sich bei der Übersicht der Technologiesets (Abbildung 62) und der Produktansicht (siehe Abbildung 63) um Darstellungen handelt, die aus der Technologieset Ansicht (Abbildung 61) generiert werden können, wurden sie nicht als eigene Softwarekarten analysiert.

Softwarekarten Beispiel

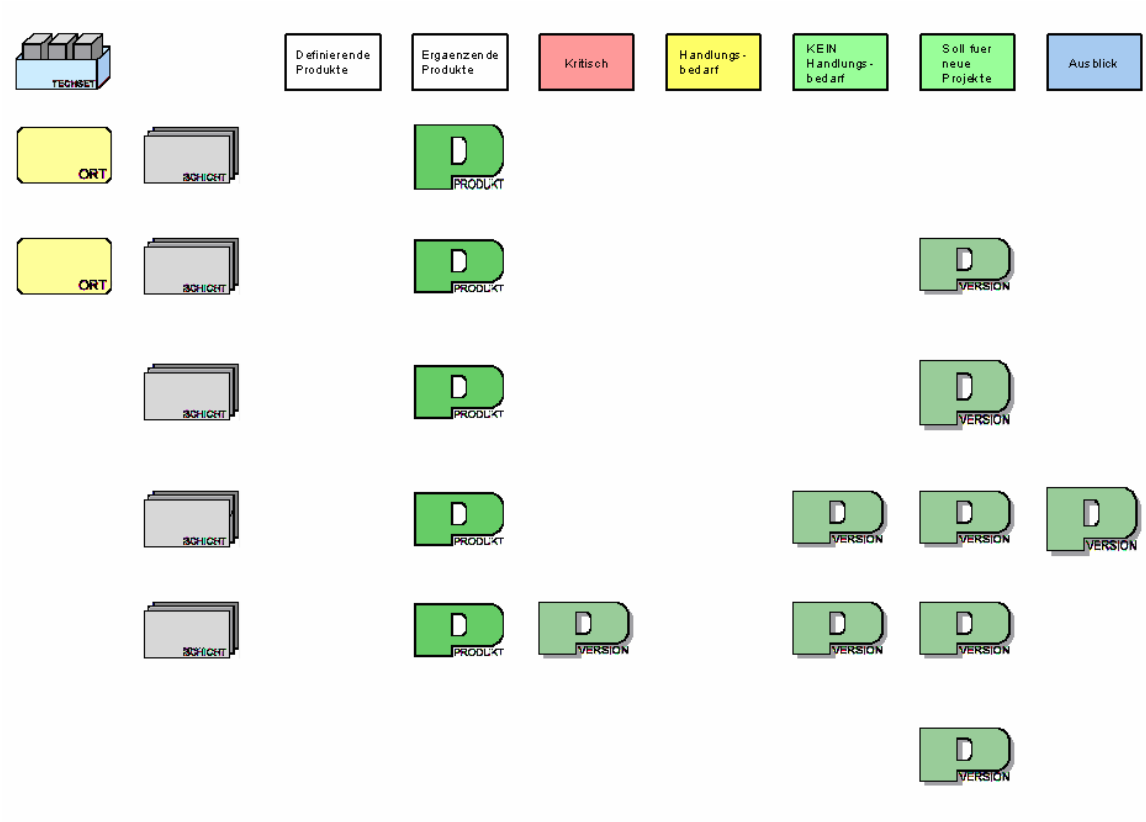


Abbildung 61: Technologieset-Karte

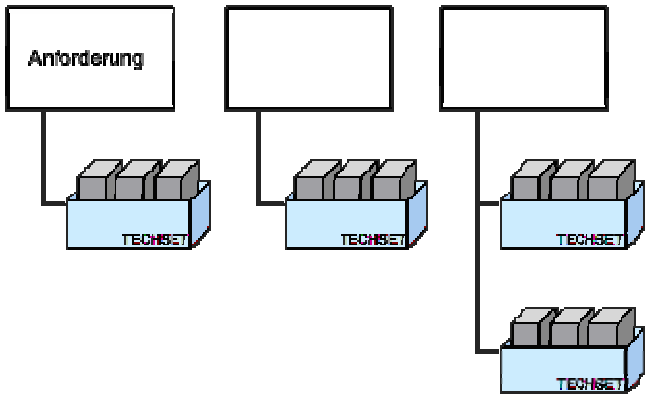


Abbildung 62: Technologieset-Karte - Übersicht der Technologiesets

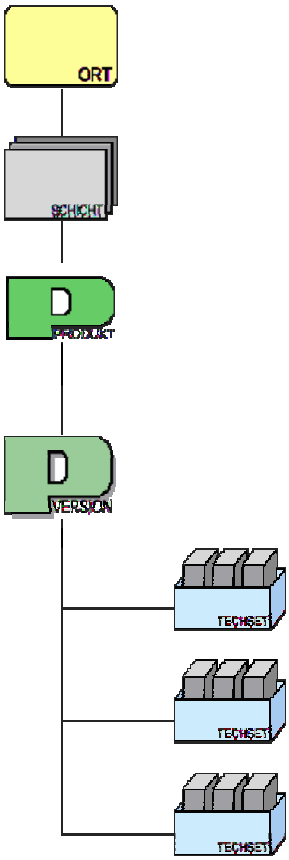


Abbildung 63: Technologieset-Karte - Übersicht der Produktverwendung in Technologiesets

Anhang B STRUKTURIERUNG EINES UNTERNEHMENS

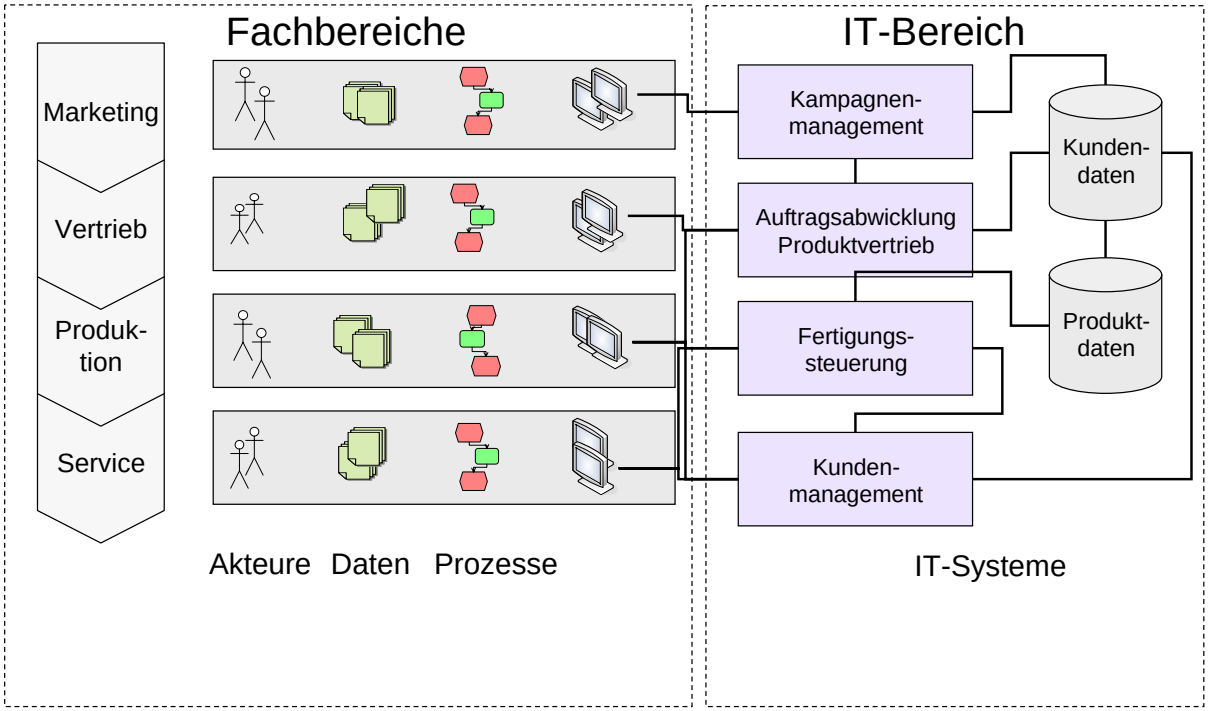


Abbildung 64: Strukturierung eines Unternehmens in Fach- und IT-Bereich aus [HKM05]

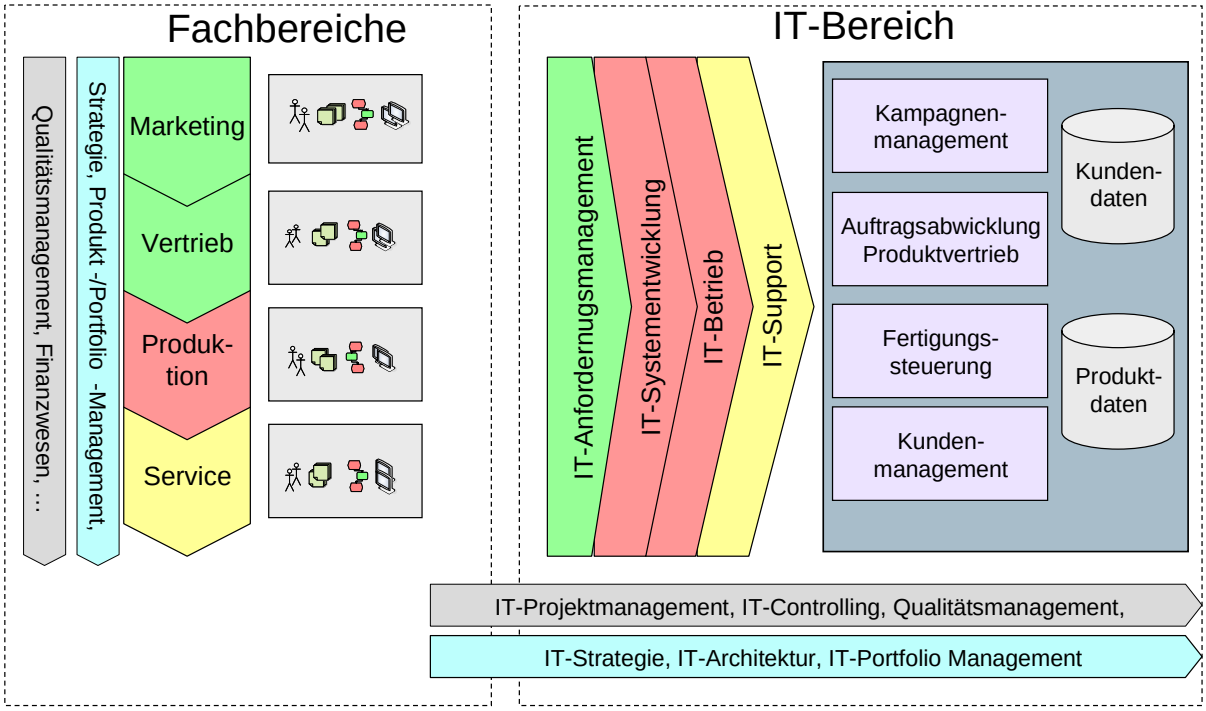


Abbildung 65: Prozesse zwischen Fachbereich und IT-Bereich aus [HKM05]

Anhang C BEGRIFFSDEFINITIONEN DES INFORMATIONSMODELLS

Begriff	Definition
Aktivität	Abstrakte Klasse die zur Generalisierung der Klassen Programm und Projekt eingeführt wurde.
Anforderung	Legt die qualitativen und quantitativen Eigenschaften oder zu erbringende Leistung eines Produkts, Systems oder Prozesses aus der Sicht des Auftraggebers fest. (in Anlehnung an [Ba00])
Anforderungsart	Charakterisiert die Art einer Anforderung, d.h. ob es sich um eine fachliche (das Was) oder eine technische (das Wie) Anforderung handelt. Ausprägungen sind die funktionale (Aussagen zu den Diensten, die das Objekt leisten sollte, zur Reaktion auf bestimmte Eingaben und zum Verhalten in bestimmten Situationen) bzw. nichtfunktionale Anforderung (Beschränkungen der durch das Objekt angebotenen Dienste oder Funktionen z.B. Zeitbeschränkungen, Standards) oder die Restriktion (engl. Constraint). [So01]
Anwendungslandschaft	Beschreibt die Aggregation von Anwendungssystemen, die dadurch charakterisiert ist, dass ihre Anwendungssysteme über verschiedene Technologien miteinander gekoppelt sein können, unterschiedliche betriebliche Geschäftsprozesse unterstützen und durch zahlreiche Projekte einem steten Wandel unterliegen. [Seb04]
Anwendungssystem	Software bzw. Software-Paket für ein geschlossenes Arbeitsgebiet, das sich logisch und technisch abgrenzen lässt und durch EDV-Funktionen (Programme) ganz oder überwiegend unterstützt wird. Die Abgrenzung zwischen verschiedenen Anwendungssystemen ist überwiegend durch fachliche Argumente gegeben, EDV-technische Argumente treten dabei in den Hintergrund. Es gibt Anwendungssysteme, die streng spartenbezogen sind (z. B. auf die Sparte Kredit oder Wertpapiergeschäft); andere Anwendungssysteme dagegen behandeln sparten-übergreifende Themen. Die Kardinalität der Assoziation zwischen Anwendungs- und Teilsystem beginnt seitens des Teilsystems mit einer null, um die Hülle für ein zu implementierendes Anwendungssystem zu definieren, ohne bereits konkrete technische Angaben und Ausprägungen zu dokumentieren. Technische Eigenschaften des Anwendungssystems werden erst in den zugehörigen Teilsystemen definiert, aus denen sich ein Anwendungssystem zusammensetzt. [HVB05b]
Architekturcontrolling	Managementkonzeption, die für die zielorientierte Steuerung der Standards und Werkzeuge der Architektur ein in sich abgestimmtes System von Prozessen, Instrumenten, Methoden und Rollen konzipiert, einführt, betreibt und weiterentwickelt. Überprüfung der Standards und Werkzeuge auf Zweckmäßigkeit, Vollständigkeit und Korrektheit. (in Anlehnung an [GA02])

Architekturmanagementprozess	Regelkreis, der für die geordnete Weiterentwicklung der Architektur im Unternehmen sorgt, indem er über klar definierte Phasen mit entsprechenden Richtlinien, Werkzeugen und Standards, IT-Komponenten konzipiert und plant. Die Einhaltung der Regeln muss dabei fortlaufend durch organisatorische Verankerung und geeignete Verfahren sicherstellt werden. [HVB05c]
Architekturpattern	Muster, das ein häufig in der Architektur wiederkehrendes Problem beschreibt. Es identifiziert und abstrahiert das Kernproblem und bietet einen allgemein gültigen Lösungsweg an. (in Anlehnung an [Ga01])
Architekturrichtlinien	Richtlinien für die Gestaltung der Architektur.
Architekturstandard	Spezifikation, d.h. ein Dokument, welches ein Produkt detailliert und präzise beschreibt, in einem öffentlichen bzw. von einer Normungsorganisation durchgeführten Verfahren angenommen wurde, sich bewährt hat und eine breite Anwendung auf dem Markt gefunden hat [RP99]. Ein Architekturstandard ist ein Standard dessen Inhalt sich architektonischen Konzepten widmet.
Aufruftyp	Nähere Klassifizierung des Aufrufs eines technischen Export-Konnektors, z.B. Call, MessageQueue, Eintrag in einer Datenbank oder direkt integrierter Code.
Bankprodukt	Eine am Markt absatzfähige Dienstleistung, die für den Kunden einen fest definierten, wahrnehmbaren Nutzen erzielt. Einem Bankprodukt können definierte Kosten zugeordnet werden (siehe IT-Management 5.2.2) und kostet einen vereinbarten Preis für den externen Kunden. Das Produkt sollte i.d.R. einen Deckungsbeitrag erzeugen und standardisierte Prozesse nutzen (siehe Geschäftsprozess-Ebene 5.2.1.2). [HVB05a]
Baustein	Systemkomponente, d.h. ein abgegrenzter Teil eines Softwaresystems, die für die physikalische Struktur einer Anwendung dient. [Ba00]
Bausteinattribut	Charakterisiert einen Baustein in Bezug auf seine Verwendung. Allgemeine Attribute beschreiben dabei nichtfunktionale Eigenschaften wie beispielsweise der Durchsatz oder die Laufzeit. Funktionale Attribute beschreiben Eigenschaften, die die Funktionalität des Bausteins detaillierter beschreiben z.B. guaranteed delivery bei Kommunikationsbausteinen. [HVB05d]
Bausteindienst	Dienste eines Integrationsbausteins, die Entwicklern und anderen Bausteinen zur Verfügung gestellt werden. Aufgrund der Dienste kann es zu Abhängigkeiten zwischen Integrationsbausteinen kommen. [HVB05d]
Bausteinfunktion	Funktionen die ein Integrationsbaustein ausführen kann. [HVB05d]
Bausteinkombination	Sinnvolle Kombination mehrerer Integrationsbausteine unter Berücksichtigung logischer Konzepte. [HVB05d]

Berechtigungssteuerungstyp	Informiert über den Zugriffskontrollmechanismus eines Anwendungssystems.
Beteiligter	Menschliche Instanz oder Rolle, die an dem Einsatz des Betriebselementes entweder als Rolle des Anbieters oder Nutzers beteiligt ist. Ein Beteiligter ist beispielsweise eine Organisationseinheit die für ihre Geschäftsprozesse Anwendungssysteme einsetzt oder ein Kunde, der den Zugang zum Firmen-Netz nutzt. (in Anlehnung an [OMG04])
Beteiligung	Interessen oder Verantwortlichkeiten, die bei einer Vereinbarung festgelegt werden und die ein Beteiligter im Bezug auf Betriebselemente besitzt. Der Einsatz eines technischen Produktes, Support Verantwortlichkeiten oder auch konkrete Dienstleistungen, die von einer Organisationseinheit erbracht werden müssen, werden beispielsweise definiert. (in Anlehnung an [OMG04])
Betriebselement	Stellvertreter aller betriebsrelevanten Objekte wie beispielsweise Produkte, Plattformen, Umgebungen oder Hardware. Für ein Betriebselement können Vereinbarungen z.B. über die Betriebszeit getroffen werden. Neben einer Beschreibung besitzt ein Betriebselement Informationen über dessen Verwaltung. Beispielsweise kann angegeben werden, welche Organisationseinheit oder welche Person angesprochen werden kann oder wo das Betriebs-Element abgeholt werden kann. (in Anlehnung an [OMG04])
Building Block	Architekturbausteine, die ähnliche Produkte bündeln, die durch gleiche oder ähnliche Geschäftsprozesse an gleiche oder ähnliche Kundengruppen vertrieben werden. Building Blocks sind aus Sicht der Anwendungen nach fachlichen Gesichtspunkten festgelegte Ausschnitte der Anwendungslandschaft, wobei *innerhalb* eines Building Blocks eine enge Kopplung der Prozesse, Kunden und Produkte besteht, während *zwischen* unterschiedlichen Building Blocks nur eine „lose“ Beziehung besteht. [HVB05e]
Codeverwaltung	Gibt Auskunft über die Verwaltung des Codes eines technischen Konnektors, z.B. Angabe der Anwendung die die Verwaltung übernimmt.
Dienst	Abstrahierte Funktion, die von einem Objekt z.B. einem Produkt bereitgestellt wird und eine in sich geschlossene Funktionskomponente aus Anwendersicht darstellt.
Dienstleistung	Immaterielles Gut, dessen Merkmal die Gleichzeitigkeit von Produktion und Verbrauch ist. [Ga00] Eine Dienstleistung wird dadurch von der Sachleistung unterschieden, als dass sie nicht lagerbar und übertragbar ist.
Dokumentiertes Wissen	Darstellung von notwendigen Informationen, Weisungen oder Standards, die z.B. in Form von Dokumenten, Zeichnungen oder Dateien dokumentiert sind. Bezieht sich ausschließlich auf solche Wissenskategorien, die explizit dokumentiert sind oder zumindest

	prinzipiell dokumentierbar wären. Ein Beispiel ist das in einem Handbuch dokumentierte Wissen über die Bedienung einer Software. Bei der Einteilung von Wissen in Wissenskategorien hilft die Unterscheidung zwischen allgemeinen Wissenskategorien und dokumentiertem Wissen dabei, Möglichkeiten und Grenzen einer Informationssystemunterstützung der Wissensverarbeitung zu identifizieren, da nur dokumentiertes Wissen elektronisch gespeichert, übertragen und verarbeitet werden kann. [IDS04]
EAI Referenzarchitektur (Enterprise Application Integration)	Methodischer Ansatz zur strukturierten, systematischen und strategischen Herangehensweise an die Integration von Anwendungen. Unterscheidet sich von konventioneller Integrationsmethodik dadurch, dass Integration in einem projektübergreifenden und strategischen Kontext betrachtet wird. Aufbau von Architekturen, in denen heterogene Anwendungen und Systeme flexibel und kostengünstig interoperieren können, ist Aufgabe der EAI. [HVB05d]
Eigenentwickeltes Produkt	Technisches Produkt, das von der Organisation selbst entwickelt wurde.
Einsatzort	Vereinbarungen können sich auf einen bestimmten Einsatzort beziehen. Dabei kann es sich bei einem Ort um ein Land, eine Stadt, eine Filiale oder sogar beispielsweise ein Gebäude handeln. Diese Angabe ist jedoch nicht immer notwendig und somit mit der Kardinalität 0..n beschriftet. Zum Beispiel kann ein Softwarelizenzvertrag geschlossen werden, ohne dass explizit definiert werden muss, in welchem Ort die Software eingesetzt wird. (in Anlehnung an [OMG04])
Enterprise-Service	Vollständiger und redundanzfreier Satz von Servicefunktionen (fachlicher Export-Konnektor), die für die Nutzung eines oder mehrerer Prozesse durch einen Servicenehmer erforderlich sind. [HVB05d]
Entscheidungsgremium	Gremien in denen Architekturentscheidungen getroffen werden, z.B. Gremien die sich mit der Architekturfreigabe von fachlichen Projekten oder der Gestaltung von Plattformen widmen. [HVB05c]
Entscheidungsrepository	Repository das alle bisherigen IT-Entscheidungen einschließlich des Datums und des entsprechenden Entscheidungsgremiums dokumentiert und archiviert.
Framework	Konfiguration von Klassen, die ein Lösungsschema für eine Problemstellung vergegenständlicht. Ein Application Framework (deutsch Anwendungsrahmen) ist eine spezielle Form eines Frameworks, das die Grundstruktur und den Steuerfluss einer vollständigen Anwendung enthält. Es liefert eine generische Lösung für eine Klasse von Anwendungssystemen, die aber in den meisten Fällen noch anwendungsspezifisch durch Ableitung spezialisiert oder durch Parametrisierung ergänzt werden muss. Darüber hinaus bieten Application Frameworks oft eine einheitliche Benutzungsschnittstelle sowohl in Form der Präsentation als auch der Handhabung. Im Vergleich mit Klassenbibliotheken ist der Steuerfluss der Anwendung insgesamt bereits durch das Application Framework definiert und wird anwendungsspezi-

	fisch nur noch angepasst. [RP99]
Fremdprodukt	Technisches Produkt, das nicht von der Organisation entwickelt, sondern von einem Hersteller bezogen wurde.
Frequenztyp	Beschreibt die Häufigkeit des Zustandekommens der Verbindung, z.B. täglich oder wöchentlich.
Geschäftsfunktion (manuell, systemunterstützt)	Geschäftsprozess bzw. dessen Teilprozess, der atomaren, transaktionsgesicherten und wieder verwendbaren Charakter besitzt. Kann auch als Business Service bezeichnet werden. Findet die Geschäftsfunktion ohne Anwendungssystem-Beteiligung statt, so handelt es sich um eine manuelle Geschäftsfunktion, ist dahingegen ein Anwendungssystem an dem Ablauf der Funktion beteiligt, so spricht man von einer systemunterstützten Geschäftsfunktion.
Geschäftsobjekt	Repräsentation eines Gegenstands aus dem realen Geschäftsleben. Die Repräsentation umfasst neben der Artbezeichnung und der Beschreibung Angaben über Attribute, Verhalten, Beziehungen und Regeln. Inhaltlich handelt es sich damit bei Geschäftsobjekten um fachliche Objekte. Typische Beispiele für Gegenstände, die durch Geschäftsobjekte dargestellt werden, sind Produkt, Rechnung, Angestellter etc. Wann und warum auf ein Geschäftobjekt zugegriffen wird, wird in Use-Cases spezifiziert. [HVB05f]
Geschäftsprozess	Zusammengehörende Abfolge von Unternehmungsverrichtungen zum Zweck einer Leistungserstellung. Ausgang und Ergebnis des Geschäftsprozesses ist eine Leistung, die von einem internen oder externen Kunden angefordert und abgenommen wird. [Sc02] Ein Geschäftsprozess lässt sich durch messbare Inputs, wertschöpfende Aktivitäten zur Erreichung von Unternehmenszielen und messbaren Outputs charakterisieren und kann Teil eines Bankproduktes sein. Geschäftsprozesse können sowohl einer Vorgänger-Nachfolger, als auch einer hierarchischen Ordnung unterliegen. Die Hierarchie von Geschäftsprozessen und ihren Teilprozessen kann dabei mittels der Attribute Prozessart und Prozessgruppe dokumentiert werden. Zu den Prozessarten zählen beispielsweise die Unternehmensmanagement Prozesse, die Kern-, Support- und die Kundengruppen und Produktionsmanagement Prozesse. Innerhalb der Prozessarten kann eine Aufteilung nach logischen Gruppen wie beispielsweise dem Aktiv- oder dem Passiv- Geschäft sowie dem Zahlungsverkehr vorgenommen werden.
Geschäftsstrategie	Grundsätzliche, langfristige Verhaltensweise (Maßnahmenkombination) der Unternehmung und relevanter Teilbereiche gegenüber ihrer Umwelt zur Verwirklichung der langfristigen Ziele. [Ga00] Die Unternehmensstrategie ist von der Geschäftsstrategie zu unterscheiden. Unter der Unternehmensstrategie versteht man die Strategie als Programm zur Sicherung künftiger Wettbewerbsvorteile im überwiegen diversifizierten Unternehmen. Die Geschäftsstrategie ist dahingegen die Strategie als Ausdruck von Produkt-Markt-Kombinationen. Die Grundlage des Wettbewerbs liefert

	bei der Unternehmensstrategie der Wettbewerb des Unternehmens im Aufbau von Kernfähigkeiten, bei der Geschäftstrategie dahingegen die Wettbewerbsfähigkeit der Produkte. [Bü99]
Hardware	Menge aller technischen Geräte einer Rechenanlage. Die Vorsilbe Hard verdeutlicht, dass es sich bei Hardware um physikalische, materielle Teile handelt [ECS93]. Es können unterschiedliche Klassifizierungen vorgenommen werden. Beispielsweise Server, Endgeräte und Netzwerkinfrastruktur Gruppierungen. Endgeräte können beispielsweise weiter zerlegt werden in Desktop und Laptop Elementen, die Netzwerkinfrastruktur in Switches, Router und Bridges.
IDL (Interface Definition Language)	Schnittstellendefinitionssprache die dazu dient, Schnittstellen von Klassen zu beschreiben, die als Server Dienstleistungen anbieten. In der IDL werden keine Programme geschrieben, sondern nur Spezifikationen, die der Anwendungsentwickler verwenden kann, um vom Client aus Operationen einer Server Klasse aufzurufen. [Ba00]
Infrastruktur	Stellt Werkzeuge und Richtlinien für eine Plattform bereit und beschreibt alles, was zum Management einer Plattform gehört.
Integrationsbaustein	Baustein der EAI Referenzarchitektur, der eine bestimmte Rolle erfüllt bzw. ein spezifisches Charakteristikum besitzt, das er über Funktionen zur Verfügung stellt. Bausteine werden gemäß ihrer Eigenschaften den vier Stufen der EAI zugeordnet, sowie einer Funktionsgruppe (z.B. Prozess-, Kommunikations-, Anbindungs- und Querschnitt). Integrationsbausteine werden produktneutral definiert, dessen resultierende Referenzarchitektur in einem aufbauenden Schritt in Form einer produktspezifischen Referenzarchitektur umgesetzt werden kann. Produkte wie beispielsweise die IBM WebSphereMQ kann als Message-Orientierte Middleware in konkreten Projekten verwendet werden um eine passende EAI zu implementieren.
Integrationspattern	Veranschaulicht die Funktionalität eines Integrationsbausteins in Form einer graphischen Repräsentation. [HVB05d]
Integrationsstufe	Um das Konzept für die Integration von Anwendungen im Unternehmen umzusetzen, eignet sich ein Satz bzw. ein Baukasten verschiedener Technologien die sich in vier verschiedene Integrationsstufen einteilen lassen: 1. *Middleware* sind die grundlegenden Integrationstechnologien, welche die Basis für die andere Stufen bzw. für die konventionelle Integration bilden 2. *Service Oriented EAI* Technologien ermöglichen dynamische Integration über einen zentralen Kommunikationsbus mit Funktionen wie dynamisches Routing, Nachrichtentransformation und Ereignissteuerung 3. *Process Oriented EAI* Technologie erlaubt die Steuerung und Kontrolle von Geschäftsprozessen über Anwendungsgrenzen hinweg

	4. *Business Oriented EAI* Technologien ergänzen prozessorientierte EAI um Prozessmonitoring und Business Intelligence Funktionalitäten. [HVB05d]
Integrationsszenario	Beschreibung einer Aufeinanderfolge von Ereignissen in Bezug auf Integrationsproblematiken. [HVB05d]
IT-Entscheidung	Eine Entscheidung die in einem Gremium über IT relevante Aspekte getroffen wurde.
IT-Managementprozess	Managementprozess, der für die geordnete Weiterentwicklung der IT im Unternehmen sorgt. Er ist Oberbegriff für das IT-Portfoliomanagement und den Architekturmanagementprozess. (in Anlehnung an [HVB05c])
IT-Portfolio	Gesamtheit der Informationssysteme eines Unternehmens, d.h. soziotechnische Systeme mit dem Ziel der optimalen Bereitstellung von Information und Kommunikation, einschließlich der Projekte zu deren Weiterentwicklung über einen mittelfristigen Planungshorizont (5-10 Jahre). [WKW94]
IT-Portfoliomanagement	Das IT-Portfoliomanagement ist ein strukturierter Management Ansatz, der das IT-Portfolio effektiv und effizient an den Unternehmenszielen, den Anforderungen der Geschäftsprozesse und den gesetzlichen Anforderungen ausrichtet. [WKW94]
IT-Projekt	Vorhaben, das im Wesentlichen durch die Einmaligkeit der Bedingungen in ihrer Gesamtheit gekennzeichnet ist, z.B. durch Zielvorgabe, durch zeitliche, finanzielle, personelle und andere Begrenzungen, durch Abgrenzungen gegenüber anderen Vorhaben und durch projektspezifische Organisation. [HVB05a]
IT-Projektreview	Review das prüft, ob im Projekt nur die erlaubten Architekturstandards eingesetzt werden und deren Einsatz nach den Architekturprinzipien stattfindet.
IT-Strategie	Strategie die durch den Einsatz von Informationstechnologie Wert für das Unternehmen schafft. Welches spezifische Wertpotential der Einsatz von IT dem Unternehmen bietet, hängt dabei, ebenso wie die Unternehmensstrategie selbst, von vielen unternehmensinternen und –externen Faktoren ab. Wichtig ist, die IT-Strategie aus der Geschäftsstrategie abzuleiten, um Voraussetzungen in der IT zu schaffen, die die Geschäftsstrategie langfristig unterstützen. Eine wertorientierte IT-Strategie muss zudem mit einem breiten Blickwinkel beginnen, der außer den unternehmensspezifischen Aspekten bei Kunden, Lieferanten und Wettbewerb auch innovative technologische Trends berücksichtigt. Globale Entwicklungen müssen einbezogen werden, um die zukünftigen Markt- und Kostenchancen des Unternehmens zu analysieren und hieraus direkte Ziele, Zielhierarchien und Anforderungen an die IT-Strategie abzuleiten. [BES04]
IT-Szenario	Stellt eine mögliche Zukunftssituation der Anwendungslandschaft dar, die unter Einfluss unterschiedlicher Veränderungen auf relevante Aspekte untersucht werden soll.

	Aspekte können dabei strategisch (z.B. Outsourcing), wirtschaftlich (Plattform muss wartbar sein), fachlich (Prozesse, Organisationseinheiten) oder auch planerisch sein, wobei der IT-Bebauungsplan die fachlichen Aspekte visualisiert, in dem die IT- Szenarien mit den zugehörigen Anwendungssystemen einschließlich der Zuordnung zu Prozessen visualisiert werden.
Kennzahl (engl. *Measures*)	Erfasst Sachverhalte quantitativ und in konzentrierter Form. Merkmale einer Kennzahl sind Informationscharakter, Quantifizierbarkeit und Informationsform. Der Informationscharakter zeigt, dass eine Beurteilung von Sachverhalten und Zusammenhängen möglich ist. Quantifizierbarkeit bedeutet, dass Sachverhalte auf metrischen Skalen gemessen werden können und dadurch "genaue" Aussagen möglich sind. Die Informationsform führt dazu, dass komplexe Sachverhalte komprimiert und auf einfache Art dargestellt werden. [Kü03]
Kommunikationsarttyp	Spezifiziert die Kommunikationsart des technischen Export-Konnektors, z.B. synchron, asynchron, publish-and-subscribe, fire-and-forget.
Komplexitätstyp	Charakterisiert die Komplexität einer Verbindung, z.B. hoch, mittel, niedrig.
Konnektor (fachlich, technisch)	Fachliche und technische Konnektoren beschreiben die Stellen von Teilsystemen bzw. Softwarekomponenten, über die eine Verbindung zwischen anbietendem und nutzendem Teilsystem abgewickelt wird. [MW04b] Konnektoren besitzen beispielsweise einen Namen, Beschreibung, Version und weitere Dokumentationsattribute, da sie beispielsweise aufgrund neuer Anforderungen oder neuer Technologien angepasst werden und dabei die Veränderungen nachvollziehbar sein müssen. Für das Abschalten einer älteren Version ist darüber hinaus ein Verfallsdatum zu definieren. Alle Nutzer einer älteren Version werden über die neue Version informiert und haben innerhalb des definierten Zeitrahmens die ältere Version abzulösen. [HVB05f] Darüber hinaus lässt sich identifizieren, ob ein Konnektor für einen Enterprise-Service relevant ist oder nicht. Fachliche Konnektoren liefern alle Angaben aus fachlicher Sicht. In ihnen wird definiert, welche Verfügbarkeit bzw. Bandbreite für die Verbindung gefordert wird, welche Vor- und Nachbedingungen bei der Funktionsnutzung eingehalten werden müssen und ob ein lesender oder schreibender Zugriff stattfindet. Ein technischer Konnektor entspricht der technisch konkret existierenden Schnittstelle und spezifiziert Implementierungsaspekte wie beispielsweise die Kommunikationsart (synchron, asynchron, publish-and-subscribe, fire-and-forget), gibt an ob die Verbindung transaktionsgesichert stattfindet, welche Signatur die Schnittstelle besitzt z.B. mittels IDL (interface definition language) beschreiben oder welche Verfügbarkeit technisch angeboten wird. Fachliche Konnektoren werden durch ein oder mehrere technische Konnektoren implementiert. Folgend sollen drei Attribute der technischen Konnektoren näher

	erläutert werden: ◆ *Behandlung Fehlercodes*: beschriebt, welche Typen an Fehlercodes (Datenbank-Fehler, Kommunikationsfehler) abgefangen werden und wie sie behandelt werden (z.B. Wegschreiben in Log-Datei, Ausgabe an Bedienerkonsole). Die Auswirkung für die Anwendungen wird verdeutlicht. Eine unterschiedliche Behandlung je nach schwere oder Typ des Fehlers ist zu beschreiben, wenn dies vorgesehen ist. Zudem kann beschrieben werden, wie ein Konnektor reagiert, wenn benötigte Daten nicht vorhanden bzw. unvollständig sind, außerhalb eines definierten Gültigkeitsbereiches liegen oder im falschen Format vorliegen. ◆ *Recovery Maßnahmen*: Beschreibt die Recovery-Maßnahmen der Schnittstelle nach einem Wideranlauf im Fehlerfall (z.B. Systemabbruch). Damit sind sowohl die für eine Recovery notwendigen Maßnahmen während des normalen Ablaufs (z.B. Checkpoints schreiben, Status verwalten) als auch die Maßnahmen beim Wideranlauf gemeint. ◆ *Betriebshinweise*: Angabe der relevanten Anforderungen wie beispielsweise Hardware (Mindesthauptspeicher, Plattenplatzanforderungen, notwendige Anbindungen), Systemsoftware (z.B. Betriebssystem, Konfigurationsparameter in Runtime-Variablen, benötigte Shell), technische Anbindung (Port xx, Socket yy) oder Netzkapazitäten. Liefert darüber hinaus Informationen über Softwarevoraussetzungen wie beispielsweise die zu verwendende Datenbank Software, genutzte Zugriffs-Libraries oder Kommunikationssoftware. Nicht zuletzt wird beschrieben, wie die Schnittstelle gestartet und gestoppt wird. Dabei sollten Name des Start- und des Stop-Scripts, Parameter der Start/Stop-Scripts, Arbeitsweise der Start/Stop-Scripts und falls bekannt wo Ablageort der Start/Stop-Scripts angegeben werden.
Konnektor (Import, Export)	Unterscheidung der fachlichen und technischen Konnektoren hinsichtlich der Exportstelle des Funktion Anbieters und der Importstelle des Funktion Nutzers. Die Export-Konnektoren sind immer dem Use-Case anbietenden Teilsystem zugehörig, z.B. bei einem lesenden Zugriff dem Teilsystem, dass die Daten an den Client liefert oder bei einem schreibenden Zugriff, dem Teilsystem, das die Daten weiterverarbeitet. Über die fachlichen Export-Konnektoren wird dabei definiert, welche Use-Cases des anbietenden Systems über die Verbindung genutzt werden können. Import-Konnektoren sind immer dem nutzenden Teilsystem zugeordnet. Export- und Import-Konnektoren werden teilweise durch gleiche Attribute beschrieben, da Import-Konnektoren eventuell nur Teile der Export-Konnektoren nutzten. Sie greifen beispielsweise nur lesend auf den Export-Konnektor zu, der sowohl lesenden als auch schreibenden Zugriff anbietet. Um den technischen Import-Konnektor des Nutzers mit dem technischen Export-Konnektors des Anbieters zu verbinden, bedarf es unter Umständen zusätzlicher Anwendungssysteme, die als Midd-

	leware dienen. (vgl. [MW04b])
Kosten	Bewerteter Verzehr von Gütern und Dienstleistungen, der durch die betriebliche Leistungserstellung verursacht wird. [Wö02]
Kostentyp	Ermöglich die genauere Spezifikation der Kosten, z.B. Bank gebuchte Kosten, Rechenzentrum gebuchte Kosten.
Kundengruppe	Zusammenfassung eine Menge von Kunden unter einer bestimmten Kategorie, z.B. Privatkunden, Firmenkunden oder Immobilien AGs. Jede Kundengruppe wird durch eine kurze Beschreibung sowie den Beziehungen zu Bankprodukten und somit auch den Geschäftsfeldern, einer bestimmten Hierarchie von Organisationseinheiten, charakterisiert.
Leistung	Wert aller erbrachten Leistungen im Rahmen der typischen betrieblichen Tätigkeit [Wö02]. Die im Informationsmodell eingeführte Klasse Leistung dient dabei als Abstraktion der beiden Klassen Dienstleistung und Sachleistung. Sie wird von einer, oder bei einer Leistung die auf hohem Abstraktionsgrad definiert wurde, von mehreren Organisationseinheiten in einer definierten Qualität, Zeit und Kosten erbracht. Werden beispielsweise Gehaltsabrechungen durch das Unternehmen auf höchster Ebene definiert und in unterschiedlichen Töchtern durch unterschiedliche Geschäftsprozesse durchgeführt. Nutzer einer Leistung kann auf der einen Seite eine Organisationseinheit des gleichen Unternehmens sein, z.B. Leistungen die eine IT-Abteilung für ihr Unternehmen erbringt. Auf der anderen Seite kann eine Leistung von einem Kunden des Unternehmens genutzt werden. Wurde zum Beispiel die IT-Abteilung als eigenständige Tochtergesellschaft ausgelagert, so nimmt die Muttergesellschaft die Rolle des Kunden ein.
Logisches Konzept	Konzepte die benötigt werden, um Integrationsbausteine, die eher rein technische Aspekte verkörpern, für ein projektspezifisches Integrationsszenario sinnvoll zu kombinieren (Bausteinkombination), z.B. Konzepte für Datenformate. [HVB05d]
Markt	Zusammentreffen von Angebot und Nachfrage, durch das sich Preise bilden. [Ga00]
Organisationseinheit	Organisatorische Einheit, die durch die Zusammenfassung und Zuordnung von (Teil-)Aufgaben zu personalen Aufgabenträgern entstanden ist. Sie erfasst die zu erfüllende Aufgabe und die zur Erfüllung dieser Aufgabe eingesetzten Menschen und Sachmittel. Gliederungspunkte für eine Organisationseinheit können dabei die Aufgaben, Themengebiete oder eine regionale Gliederung sein. Organisationseinheiten bzw. Geschäftsfelder sind an der Ausführung von Geschäftsprozessen beteiligt bzw. verantwortlich. Organisationseinheiten werden meist hierarchisch gebildet, so dass eine Organisationseinheit meist eine übergeordnete Organisationseinheit besitzt, der sie disziplinarisch unterstellt ist. Einer Organisationseinheit wird mindestens ein Standort zugeord-

	net. [Bü99]
Organistationseinheitstyp	Legt für eine Organisationseinheit fest, ob es sich beispielsweise um eine Abteilung, Filiale, Niederlassung oder ein Unternehmen wie beispielsweise die HVB Info oder HVB Systems handelt.
Ort	Klassifikation von Produkten innerhalb der Technologiesets.
Person	Gesamtheit aller Eigenschaften eines Menschen.
Plattform	Technologische Ausprägung eines oder mehrerer Bausteine. Sie umfasst Produkte, Prozesse (z.B. Implementierung, Deployment) und steht im Zusammenhang mit einer bestimmten Infrastruktur und einer Umgebung (z.B. Entwicklungsumgebung). Eine Basis-Plattform stellt Basisdienste zum Betrieb der erweiterten Plattformen zur Verfügung. Zur Basis-Plattform gehören die Betriebssystem- und die Hardware-Ebene. Beispielplattformen sind Host (RACF, DB2, TSO), Windows NT 4.0, Unix, SAP/R3.
Produktfunktion	Funktionen die ein technisches Produkt zur Erbringung eines Dienstes anbietet.
Produktpaket	Zusammenfassung mehrerer Produkten gemäß definierter Aspekte. Beispielsweise können Produkte, die eine Plattform implementieren zu einem Produktpaket gebündelt werden.
Produktversion	Version eines technischen Produkts, die identifiziert, definiert und als eigenständiges Package ausgehändigt werden kann.
Programm	Laut CCTA (Central Computer and Telecommunications Agency) versteht man unter einem Programm: "A portfolio of projects to achieve a set of business objectives". [OGC05]
Protokolltyp	Dokumentiert das für eine Verbindung verwendete Protokoll eines technischen Konntektors, z.B. TCP/IP oder SOAP.
QoS Typ	Angabe der Quality of Service Eigenschaften von Konnektoren, z.B. guaranteed Delivery, once and only once, Transaktionsmanagement, sequencing oder Recovery fähig.
Qualifikation	Individuelles Arbeitsvermögen, d.h. die Gesamtheit der subjektiv-individuellen Fähigkeiten, Kenntnisse und Verhaltensmuster, die es dem einzelnen erlauben, die Anforderungen in bestimmten Arbeitsfunktionen auf Dauer zu erfüllen. [Ga00]
Referenzarchitektur	Muster (engl. Pattern), welches ein in einer bestimmten Umgebung beständig wiederkehrendes Problem beschreibt und den Kern dessen Lösung erläutert. Eine Lösung kann auf diese Weise für gleichartige Probleme beliebig oft angewendet werden. [Ga01] Eine Referenzarchitektur kombiniert insbesondere Architekturpatterns von Bausteinen aus mehreren Schichten (vgl. System-Ebene).
Releasarttyp	Informationen über den Release-Status eines Anwendungs-, Teilsystems oder einer Sotwarekomponente, z.B. Angaben über die

	Umgebung für das Management des Releasezyklus.
Releasestatustyp	Anwendungssysteme, Teilsysteme und Softwarekomponenten werden über einen Release-Status näher beschrieben. Sie befinden sich entweder in Planung, Entwicklung, Test, Einsatz oder sind bereits archiviert.
Review	Unter einem Review versteht man die manuelle, semiformale Prüfmethode, um Stärken und Schwächen eines schriftlichen Dokuments anhand von Referenzunterlagen zu identifizieren und durch den Autor beheben zu lassen. [Ba98]
Risikoklassetyp	Ausprägung des Attributs Risikoklasse von Anwendungs- bzw. Teilsystemen, wobei das Teilsystem die Risikoklasse des Anwendungssystems erben kann. Ausprägungen können wie folgt sein: *äußerst kritisch*, *sehr kritisch* (default), *kritisch* oder *weniger kritisch*. Ein Anwendungssystem wird beispielsweise als *sehr kritisch* eingestuft, wenn bei dessen Ausfall eine Gefährdung des ordentlichen Geschäftsbetriebes nach ca. einem Tag auftritt und Bankkunden oder andere externe Partner betroffen sind. Es hat sich gezeigt, dass die Risikoklasse in vielen Fällen nicht die Vereinbarung mit dem Kunden widerspiegelt. Daher dokumentiert das Attribut Risikoklasse Kunde die Vereinbarung mit dem Kunden. Unterschiedliche Werte in "Risikoklasse" und "Risikobetrachtung des Kunden" zeigen nun die unterschiedlichen Sichten des Systemverantwortlichen und des Kunden auf. Es ist beabsichtigt, dass diese beiden Werte in naher Zukunft übereinstimmen. Die Vererbung und der Wertebereich des Feld "Risikobetrachtung des Kunden" entsprechen denjenigen der "Risikoklasse". [HVB05b]
Rolle	Definiert alle für die Stelle benötigten Qualifikationen, die von der die Rolle besetzenden Person erfüllt werden sollten. [Bü99] Darüber hinaus fasst sie alle Fähigkeiten, Verantwortungen und Aufgaben unter einem Namen, um unabhängig von Personen über diese „Rolle“ sprechen zu können. [GA02] Eine Person kann verschiedene Rollen innehaben, und eine einzige Rolle kann mit mehreren Mitarbeitern verbunden sein. [So01]
Sachleistung	Leistung, die nicht den Dienst- oder Geldleistungen zuzuordnen sind. (in Ahnlehnung an [Wö02])
Schicht	Strukturierungsmerkmal einer Schichtenarchitektur und meist dadurch gekennzeichnet, dass Komponenten innerhalb einer Schicht beliebig aufeinander zugreifen können, Komponenten zwischen unterschiedlichen jedoch nur gemäß strenger Zugriffsregeln. [Ba00]
Servicedomäne	Verwaltungseinheit, die sich über eine Menge von Services definiert und immer einem bestimmten Geschäftsfeld zugeordnet ist. Eine Servicedomäne kann ein Building Block sein, der anderen Building Blocks seine Servicekomponenten zur Verfügung stellt. Generell ist jede Servicedomäne für die Pflege seiner Geschäftsobjekte und für die Bereitstellung der Interfaces zu anderen Servi-

	cedomänen zuständig. Sie besitzt u.a. Mechanismen um Enterprise-Services zu erstellen, Festlegungen für die Formate der Enterprise-Services und ihre Lokationsorte sowie Verfahren für die Kapselung von Business-Logik. Eine Servicedomäne muss nicht immer genau einem Building Block zuordbar sein, sondern kann sich auch über mehrere Building Blocks ausdehnen. Jede Servicedomäne (z.B. Building Block, Geschäftseinheit) stellt ihre eigenen Enterprise-Services zur Verfügung und hat Kontrolle darüber, welche Enterprise-Services durch ihre Einheit angeboten werden. Die Servicedomäne ist der einzige in das Informationsmodell zusätzlich einzuführende SOA Begriff, alle anderen konnten Klassen des Informationsmodells zugeordnet werden oder besitzen einen Abstraktionsgrad, der nicht im Informationsmodell erfasst werden soll, z.B. Legacy System, Intermediary. [HVB05d]
SOA Referenzarchitektur (serviceorientierte Architektur)	"…a best-practice architecture pattern for the systematic design of request/reply applications. Its primary intentions are business-level software modularity and rapid, nonintrusive reuse of business software in new runtime contexts. Users must understand the essence of SOA, as well as its strength and limitations, to identify its role in the overall architecture of modern enterprise IT." [NS03] Die SOA Referenzarchitektur kann auf der Integrations-Ebene angeordnet werden, da sie auf EAI und auf der Anwendungssystem-Ebene fachliche Funktionalitäten zur Verfügung stellt. [HVB05d] Die SOA besteht aus grobgranularen Softwareeinheiten, die geschäftlich-fachliche Funktionen anbieten und mit anderen Anwendungen und Services durch ein lose gekoppeltes, nachrichtenbasiertes Modell kommuniziert. In einer SOA sind Anwendungen als eine Menge voneinander unabhängigen Services zu implementieren, die wohldefinierte Schnittstellen zur Verfügung stellen. Demzufolge handelt es sich bei der SOA um eine Softwarearchitektur, bei der Funktionen mit bestimmten Eigenschaften auf eine definierte Art für Servicenehmer zur Verfügung gestellt werden. [HVB04d]
Softwarearchitektur-Bebauungsplan	Ermöglicht in Form eines Baukastens für Softwarearchitekturkomponenten (Bausteine) die Klassifizierung von Produkten anhand ihrer Funktionalität, unterstützt die Produkt-Standardisierung und dient der Identifikation fehlender Bausteine bzw. Produkte.
Softwareeinheit	Abstrahiert die Klassen Softwarekomponenten und Softwareelemente und bietet den Angriffspunkt für eine bis in kleinste Detail vorgenommene Softwarebeschreibung die zukünftig von der HVB Systems im Firmen-Repository vorgenommen werden soll. Bei der Beschreibung von Verbindungen zwischen Teilsystemen, die mittels Software-Komponenten realisiert werden ist im Allgemeinen zwischen Verbindungen zu Komponenten anderer Teilsysteme (externe) oder zu Komponenten des gleichen Teilsystem (interne) zu unterschieden. Die Art der Beschreibung dieser Teilsystem "extern und internen" Verbindungen unterscheidet sich nicht, jedoch sind Interne im Hinblick auf die Beschreibung von Anwendungslandschaften nicht von Interesse.

Softwareelement	Bestandteil von Softwarekomponenten, d.h. identifizierbare, kleinste als unteilbar behandelte Bestandteile des Teilsystems, das Änderungen unterworfen ist. Innerhalb der J2EE Entwicklungsumgebung ist ein Software-Element ein so genanntes Item, das in vielen Fällen einer Datei am Filesystem entspricht.
Softwareelementtyp	Klassifiziert unterschiedliche Softwareelemente, z.B. kann ein Softwareelement im J2EE Umfeld einem so genannten Item entsprechen. [HVB04a]
Softwarekomponente	Fortsetzung der hierarchischen Beziehung von Anwendung- und Teilsystemen. Entspricht einer logischen Einheit die hochqualitativ, wieder verwendbar, gut skalierbar ist und technische Konnektoren realisiert bzw. besitzen. Ein Beispiel für Komponenten sind im J2EE Umfeld die Module. Überlegungen die bei der Schneidung solcher dienen sind beispielsweise die Größe eines Moduls (Übersichtlichkeit vs. Wartbarkeit), fachliche bzw. technische Zusammengehörigkeit, Reifegrad der Modulelemente (siehe Software-Element) im Bezug auf das Release-Management und die Skalierbarkeit. [HVB04a]
Softwarekomponententyp	Klassifizierung der Softwarekomponente, z.B. Host Komponente (Programm, Modul) vom Typ: PGMCOB, MODCOB, PGMASM, MODASM, PGMTLN, PGMEARL, PGMPLI, PGMOPA, OBJC oder PGMC handeln.
Standort	Geographischer Ort, an dem Produktionsfaktoren zur Erstellung betrieblicher Leistungen eingesetzt werden. [Wö02]
Stelle	Kleinste organisatorisch zu definierende Organisationseinheit, die durch Zuordnung von (Teil-)Aufgaben und gegebenenfalls von Sachmitteln auf einen einzelnen menschlichen Aufgabenträger entsteht. Eine Stelle ist weder räumlich noch zeitlich festgelegt und wird in der Regel personenunabhängig definiert, damit sie von der Aufgabe her determiniert und von einem Stellenwechsel einer konkreten Person nicht bedroht ist. Aus diesem Grund wird der Stelle über das so genannte Rollenkonzept eine Person zugeordnet. [Bü99]
Systemtyp	Charakterisiert den Typ eines Anwendungs- bzw. Teilsystemens, z.B. Kernsystem.
Technisches Produkt	Produkt, das für die Entwicklung oder den Einsatz von Anwendungssystemen verwendet wird. Es bietet über seine Produktfunktionen bestimmte Dienste an und wird in einem Paket ausgeliefert. Ein Produkt kann ein Fremdprodukt oder ein eigen entwickeltes Produkt sein.
Technologie	Gesamtheit der anwendbaren und tatsächlich angewendeten Arbeits-, Entwicklungs-, Produktions- und Implementierungsverfahren der Technik, z.B. .NET. [RP99]
Technologieset	Beschreibt eine für die Entwicklung sowie den Betrieb von Anwendungssystemen einsetzbare, gültige Kombination von technischen Produkten inklusive Versionsnummern und ihren aktuellen Le-

	benszyklus-Status. Die im Technologieset angegebene Kombination von technischen Produkten und Versionsnummern stellt eine zusammenpassende und in dieser Kombination auch sinnvolle Einheit dar. Der im Technologieset abgebildete Lebenszyklus ermöglicht es den Gültigkeitszeitraum von Produktkombinationen zu managen und somit frühzeitig auf Veränderungen zu reagieren. [HVB05c]
Technologiesetart	Dient der näheren Klassifikation eines Technologiesets, z.B. Front- oder Back-Office Technologieset.
Technologiesettyp	Dient der näheren Klassifikation eines Technologiesets, z.B. Technologieset für Internet oder Host Programme.
Technologietrend	Entwicklung der Technologie über die Zeit. (in Anlehnung an [Wö02])
Teilsystem	Bestandteil eines Anwendungssystems, das in mehrerer Hinsicht einfacher zu handhaben ist, z.B. Überschaubarkeit aus technischer und fachlicher Sicht. Ein Beispiel ist das Anwendungssystem 'Depotverwaltung' mit den Teilsystemen Depoteröffnung und –Änderung, Bewegungen, Kauf/Verkauf und Auswertungen und Statistik. Teilsysteme werden in verschiedene Typen unterteilt, wobei im Wesentlichen die Plattformen unterschieden werden, z.B. Teilsystem NT (=NT-Produkte) definiert die eigen entwickelten Teilsysteme, die NT als Plattform haben wohingegen Teilsystem NT-FSW (=NT-Produkte) die Fremdsoftware-Teilsysteme definiert, die NT als Plattform haben. Ein Teilsystem kann Technische Produkte (siehe 5.2.1.5) verwenden und läuft auf bestimmten Hardwareelementen. Um bei dem Anlegen eines neuen Anwendungssystems nicht zwingend die Definition der benötigten Hardware vorzuschreiben, wurde hier seitens der Hardware die Kardinalität 0..n gewählt. [HVB05b]
Teilsystemtyp	Teilsysteme können durch Angabe ihres Typs näher spezifiziert werden, z.B. Teilsystem Host, Unix, NT oder Java bzw. Teilsystem Fremdsoftware Host, Unix, NT oder Java.
Umgebung	Stellt Methoden und Verfahren, Anforderungen für Werkzeuge, Richtlinien, Testwerkzeuge usw. bereit und beschreibt alles, was zum Management einer Umgebung gehört, z.B. Entwicklungsumgebung, Integrationsumgebung, Qualitätssicherungsumgebung und Produktion.
Unternehmen	Zeichnet sich durch einen wirtschaftlichen Zweck und seiner Unternehmensziele aus. [Bü99]
Unternehmensziel	Formalziele wie z.B. Gewinn und Umsatz [Bü99] Die Zielbildung und –Konkretisierung ist eine sehr wichtige Managementfunktion: „Ziele sind Maßstäbe, an denen künftiges Handeln gemessen werden kann. Man braucht Ziele, um sagen zu können, wie gut oder schlecht Aktionen sind und inwieweit sich Aktionen unterscheiden. Man braucht Ziele, um optimale Aktionen zu erkennen: Optimale Entscheidungen sind eben Ziel entspre-

	chende Entscheidungen.“ [Wö02]. Beispiele für Unternehmensziele sind die "Erhöhung der Beratungskompetenz" oder die "Kundenorientierung leben".
URL (uniform resource locator)	Im Web verwendete standardisierte Darstellung von Internetadressen. [Ba00]
Use-Case (Anwendungsfall, Geschäftsvorfall, IT-Service)	Kapselung von einer in sich geschlossenen Sammlung aller Aktionen, die in einer spezifizierten Reihenfolge ablaufen. [Oe04] Die Beziehung von Use-Cases und systemunterstützten Geschäftsfunktionen ist keine eins-zu-eins Beziehung, da auch mehrere Use-Cases für die Durchführung einer systemunterstützten Geschäftsfunktion notwendig sein können.
Verbindung	Benötigt ein Anwendungssystem zur Erbringung seiner fachlichen Funktion die Unterstützung eines anderen Anwendungssystems, so geht es eine Verbindung ein. Ein Anwendungssystem nimmt dabei die Rolle des Use-Case Nutzers, das andere die des Use-Case Anbieters ein. Das Verbindungs-Attribut Verbindungstyp dokumentiert, ob es sich dabei um eine Funktionsnutzung oder einen Datenfluss handelt. Die Eigenschaft Frequenz gibt an, ob die Verbindung zwischen den Teilsystemen täglich, wöchentlich oder jährlich zustande kommt. Häufig bedarf es einer Klassifikation der Verbindung hinsichtlich ihrer Komplexität, so dass der Aufwand, der bei der Erstellung dieser Verbindung notwendig ist eingeschätzt werden kann. Die Problematik liegt jedoch darin, geeignete Parameter zu definieren, die die Einstufung in eine hohe, mittlere oder niedrigere Komplexitätsklasse möglichst objektiv erlaubt. Mögliche Kriterien könnten die Anzahl der zu berücksichtigenden Softwarekomponenten sein, Umfang benötigter Middleware oder die Qualität der vorliegenden Schnittstellendokumentation. (vgl. [MW04b])
Verbindungstyp	Angabe, ob es sich bei einer Verbindung um eine Funktionsnutzung oder einen Datenfluss handelt.
Vereinbarung	Eingegangene Verbindlichkeit, ein geschlossener Vertrag mehrerer Beteiligter unter Angaben der zu leistenden Beteiligung und des betreffenden Betriebselementes. Im Speziellen kann eine Vereinbarung beispielsweise ein Vertrag für Software Lizenzen, ein Support Vereinbarungen oder ein Service Level Agreement sein, d.h. ein Leistungsschein über eine Vereinbarung zwischen einem Dienstleister und seinem Kunden einschließlich der zu erbringenden Dienstleistung. (in Anlehung an [OMG04])
Verfügbarkeitstyp	Angabe einer konkrete Zeit für die Verfügbarkeit eines Konnektors, z.B. 24 Stunden pro Tag.
Vorfall (engl. Incident)	Dient der Bearbeitung und Verfolgung von Störungsmeldungen, die durch Anwender gemeldet werden. Erstes Ziel bei der Bearbeitung von Incidents besteht darin, dem Anwender die Weiterarbeit mit der betroffenen Anwendung, gegebenenfalls durch eine Um-

	gehungslösung, zu ermöglichen. [HVB04e]
Wissenkategorie	Dient der thematischen Klassifizierung von Wissensinhalten ohne entsprechende Dokumentation. Beispiele für Wissenskategorien sind: „… Wissen über eine bestimmte Branche, Wissen über eine bestimmte Technologie,...“. Bei dem in eine bestimmte Wissenskategorie eingeteilten Wissen kann es sich sowohl um implizites Wissen handeln, also um nicht vollständig dokumentierbares, als Fähigkeiten von Mitarbeitern oder Gruppen vorhandenes Wissen, als auch um explizites Wissen, das etwa in Form einer Beschreibung oder einer technischen Zeichnung dokumentiert sein kann. Häufig beinhalten Wissenskategorien beides. [IDS04]
Zugriffsarttyp	Gibt bei den technischen Konnektoren die geforderte bzw. die angebotene Zugriffsart an, z.B. lesend oder schreibend.
Zugriffsoptiontyp	Gibt für den technischen Export-Konnektor die Art der Kommunikationssteuerung an, z.B. online, offline oder manuell.
Zukunftsaussichttyp	Charakterisiert ein Anwendungssystem, z.B. wird ausgebaut, bleibt unverändert, wird abgelöst durch Projekt, wird abgeschaltet, wird wahrscheinlich abgeschaltet durch Projekt, wird wahrscheinlich abgeschaltet.

Tabelle 10: Begriffsdefinitionen des Informationsmodells

Anhang D LITERATURVERZEICHNIS

[Ba00] Balzert, Helmut: *Lehrbuch der Software-Technik – Software-Entwicklung*. 2. Auflage, Spektrum Akademischer Verlag, Heidelberg, Berlin, 2000, ISBN 3-8274-0480-0.

[Ba98] Balzert, Helmut: *Lehrbuch der Software-Technik – Software-Management, Software-Qualitätssicherung, Unternehmensmodellierung*. Spektrum Akademischer Verlag, Heidelberg, Berlin, 1998, ISBN 3-8274-0065-1.

[Baf05] Bundesanstalt für Finanzdienstleistungsaufsicht: *Aufgaben und Ziele*. <http://www.bafin.de/bafin/aufgabenundziele.htm> (abgerufen 2005-05-18).

[BBB03] Bernhard, Martin G.; Blomer, Roland; Bonn, Jürgen: *Strategisches IT-Management: Organisation – Prozesse – Referenzmodelle.* Band 1, Symposion Publishing, Düsseldorf, 2003, ISBN 3-936608-34-2.

[Be00] Becker, Jörg; Schütte, Reinhard; Geib, Thomas; Ibershoff, Hartmut: *Grundsätze ordnungsmäßiger Modellierung (GoM)*. Sachbericht, 2000-03-23; <http://www.wi.uni-muenster.de> (abgerufen 2005-07-03).

[Be04] Beyer, Nico: *Kennzahlen zur Beschreibung von Anwendungslandschaften und ihre Visualisierung auf Softwarekarten*. Bachelorarbeit, Technische Universität München, Fakultät für Informatik, 2004.

[BES04] Buchta, Dirk; Eul, Marcus; Schulte-Croonenberg, Helmut: *Strategisches IT-Management: Wert steigern, Leistung steuern, Kosten senken*. 1. Auflage, Gabler Verlag, Wiesbaden, 2004, ISBN 3-8349-0007-9.

[BH04] Buhl, Ulrich; Heinrich, Bernd: *Unternehmensarchitekturen in der Praxis – Architekturdesign am Reißbrett vs. Situationsbedingte Realisierung von Informationssystemen*. In: Wirtschaftsinformatik 46, 2004.

[BHZ04] Brenner, Walter; Hochstein, Axel; Zarnekow, Rüdiger: *ITIL als Common-Practice-Referenzmodell für das IT-Service Management*. In: Wirtschaftsinformatik 46, 2004.

[Bo05] Bonsangue, Marcello; Buuren, René; Jonkers, Henk; Lankhorst, Marc; Hoppenbrouwers, Stijn; Van der Torre, Leendert: *Concepts for Modelling Enterprise Architectures*. Information Centre of Telematica Instutuut, Niederlande, Technischer Bericht, 2005.

[BP01] Buhl, Christ; Pape, Ulrich: *Marktstudie: Software Systeme für Enterprise Application Integration (EAI)*. ALB/HNI-Verlagsschriftreihe, Paderborn, 2001.

[BRS95] Becker, Jörg; Rosemann, Michael; Schütte, Reinhard: *Grundsätze ordnungsmäßiger Modellierung*. In: Wirtschaftsinformatik 37, 1995.

[BS96] Becker, Jörg; Schütte, Reinhard: *Handelsinformationssysteme*. Verlag Moderne Industrie, Landsberg, Lech, 1996.

[Bü05] Büchner, Thomas: *Introspektion – Eine Methodik zur Entwicklung introspektiver Software-Architekturen*. Technische Universität München, Fakultät für Informatik, Lehrstuhl für Informatik 19 (sebis), Forschungsprojekt Introspektion; <http://wwwmatthes.in.tum.de> (abgerufen am 2005-06-02).

[Bü99] Bühner, Rolf: *Betriebswirtschaftliche Organisationslehre*. 9. Auflage, Oldenbourg Verlag, München, Wien, 1999, ISBN 3-486-25096-5.

[BV03] Berenz, Bernd; Voit, Franz: *Die Geschäftsprozessorientierung in der Abschlussprüfung*. In: Die Wirtschaftsprüfung 22, 2003.

[CCI05] Kompetenzzentrum Integration Factory (CC IF): *Nachhaltige Gestaltung und Bewirtschaftung heterogener Applikationslandschaften.* Universität St. Gallen, Institut für Wirtschaftsinformatik, Juni 2004 – Mai 2006; <http://ccif.iwi.unisg.ch/konzept.php> (abgerufen 2005-12-18).

[DMT05] Distributed Management Task Force: *CIM Tutorial*. <http://www.wbemsolutions.com/tutorials/CIM/index.html> (abgerufen 2005-05-17).

[Ec05] Eckert, Sven, Ferstl, Otto K.; Isselhorst, Tilman; Sinz, Elmar J.: *eEconomy, eGovernment, eSociety.* In: 7. Internationale Tagung Wirtschaftsinformatik 2005, Bamberg, Germany, 2005.

[ECS93] Engesser, Hermann; Claus, Volker; Schwill, Andreas: *Duden Informatik - Ein Sachlexikon für Studium und Praxis*. Dudenverlag, Mannheim, 1993, ISBN 3-411-05232-5

[EL04] McLean, Ephraim R.; Luftman, Jerry: *Key Issues For IT Executives*. In: MIS Quarterly Executive 3 (2), University of Minnesota, 2004.

[Enc04] Encyclopaedia Britannica Online: *Architecture*. <http://www.britannica.com> (abgerufen 2004-12-16).

[Es02] Esser, Manfred: *Komplexitätsbeherrschung in dynamischen Diskurswelten – Ein Metamodell zur Modellierung betrieblicher Informationssysteme*. Dissertation, Reihe: Wirtschaftsinformatik, Band 41, Josef Eul Verlag, Lohmar, Köln, 2002, ISBN 3-89936-036-2.

[Ga00] Gabler: *Wirtschaftslexikon*. 15. Auflage, Gabler Verlag, Wiesbaden, 2000, ISBN 340930388X.

[Ga01] Gamma, Erich; Helm, Richard; Johnson, Ralph; Vlissides, John: *Entwurfsmuster – Elemente wieder verwendbarer objektorientierter Software*. Addison-Wesley Verlag, 2004, ISBN 0-201-63361-2.

[GA02] Gernert, Christiane; Ahrend, Norbert: *IT-Management: System statt Chaos – ein praxisorientiertes Vorgehensmodell*. 2. Auflage, Oldenbourg Verlag, 2002, ISBN 3-486-25939-3.

[He02] Heinrich, Lutz J.: *Informationsmanagement.* 7. Auflage, Oldenbourg Verlag, München, Wien, 2002, ISBN 3-486-25842-7.

[HGM02] Hake, Günter; Grünreich, Dietmar; Meng, Liqiu: *Kartographie: Visualisierung raum-zeitlicher Informationen*. 8. Auflage, De Gruyter Verlag, 2002, ISBN 3-11-016404-3.

[HKM05] Helbig, Johannes; Kempf, Peter; Matthes, Florian: *Strategisches IT-Management – Motivation, Rahmen und Einführung*. Vorlesungsunterlagen der Technischen Universität München, Fakultät für Informatik, Sommersemester 2005.

[HVB03a] Penzel, Hans: *Architekturmanagement aus Sicht einer Grossbank*. HVB Systems, 2003.

[HVB03b] Schmid, Alexander; Luber, Adalbert: *Enterprise Architekturen – Best Practices und Erfahrungen aus der praktischen Anwendung*. Version 1.1, HVB Systems, 2003-12-01.

[HVB03c] Stirmlinger, Herbert: *Wie finde ich mein Technologieset?*. HVB Systems, München, 2004-05-08.

[HVB04a] HVB Systems: *J2EE - Versionierung Übersicht*. HVB Systems; <http://eu.intranet.hypovereinsbank.de/eu/j2ee/index.htm> (abgerufen am 2004-11-08).

[HVB04b] HVB Systems: *SYS30SW im Überblick*. Schulungsvortrag, Version 1.0, München, 2004-01-09.

[HVB04c] Krump, Oliver: *Projektmanagement im IT-Umfeld der HVB Group*. Version 4.0, HVB Systems,2004.

[HVB04d] Popescu, Gino-Alexandre; Röwekamp, Peter: *SOA-Glossar*. Version 1.0, HVB Systems, 2004-09-06.

[HVB04e] Stamer, Andreas: *Einzelnachweis von Aktivitäten zur Produktionssicherung*. Version 1.4., Benutzerhandbuch, HVB Systems, 2004.

[HVB05a] HVB Group: *Plus-Produktkatalog*. <http://info.intranet.hypovereinsbank.de/io-cms/krc/vertrieb/1680.html> (abgerufen am 2005-05-24).

[HVB05b] HVB Systems: *Glossar*. Glossar des Firmen Repositorys, 2005.

[HVB05c] HVB Systems: *Architektur im Überblick – Architektursteuerung, Bebauungspläne und Architektur im Projekt*. Schulungsvortrag, Version 1.15, München, 2004.

[HVB05d] Ackermann, Frank; Aspeck, Günther; Drexelius, Klaus; Haring, Roland; Keuntje, Jan Holger; Konrad, Hermann; Popescu, Gino-Alexandre, Röwekamp, Peter; Scharnetzke, Marco; Taeger, Lars: *HVB-EAI-Referenzarchitektur.* Version 2.6, HVB Systems, 7.4.2005.

[HVB05e] Ehinger, Peter; Hess, Andreas; von der Ropp, Georg: *Architekturprinzipien für die Gestaltung von Schnittstellen*. Version 1.0, HVB Group, 2005.

[HVB05f] Steinheimer, Klaus; Xu, Chengmao; Mehls, Kai-Uwe; Popescu, Gino: *Architekturprinzipien für die Integrationsinfrastruktur*. Version 0.3, HVB Systems, München, 2005-02-28.

[IDS04] IDS Scheer: *ARIS Methode - Handbuch zur ARIS 6 Collaborative Suite.* 2004.

[IE00] IEEE: *IEEE Std 1471-2000: Recommended Practice for Architectural Description of Software-Intensive Systems*. IEEE Computer Society, 2000.

[Ja03] James, Great A.: *High-Level Architecture Models Give Enterprise Perspective*. Gartner Research, ID Number: TU-21-1018, 2003-11-17.

[Ju04] Jung, Elke: *Ein unternehmensweites IT-Architekturmodell als erfolgreiches Bindeglied zwischen der Unternehmensstrategie und dem operativen Bankgeschäft*. In: Wirtschaftsinformatik 46, 2004.

[KE01] Kemper, Alfons; Eickle, André: *Datenbanksysteme – Eine Einführung*. 4. Auflage, Oldenbourg Verlag, München, 2001, ISBN 3-486-25706-4.

[KK05] Knox, Mary; Kreizman, Gregg: *Enterprise Architecture Varies By Industry.* Gartner Research, ID Number: G00126614, 2005-03-28.

[KKS04] Klein, R.; Kupsch, F.; Scheer, August-W.: *Modellierung inter-organisationaler Prozesse mit Ereignisgesteuerten Prozessketten*. In: Veröffentlichungen des Instituts für Wirtschaftsinformatik, Heft 178, Saarbrücken, 2004, <http://iwi.uni-sb.de/Download/iwihefte/iwiheft_178.pdf> (abgerufen am 2005-01-05).

[Kn02] Knox, Mary: *Enterprise Architecture: The Enabler of Success*. Gartner Research, ID Number: AV-18-0146, 2002-09-05.

[Kn04] Knox, Mary: *Alignment Is Enterprise Architecture's Critical Mission*. Gartner Research, ID Number: AV-22-0076, 2004-01-27.

[Ko04] Kohlstock, Peter: *Kartographie – Eine Einführung*. Ferdinand Schöningh Verlag, Paderborn, 2004, ISBN 3-506-71710-3.

[Kü03] Kütz, Martin.: *Kennzahlen in der IT - Werkzeuge für Controlling und Management*. 1. Auflage, dpunkt.verlag, Heidelberg, 2003; ISBN 3-89864-225-9.

[La04] Lapkin, Anne: *A User's Guide to Architectural Patterns*. Gartner Research, ID Number: G00124049, 2004-11-22.

[La05a] Lapkin, Anne: *The Seven Fatal Mistakes of Enterprise Achitecture*. Gartner Research, ID Number: G00126144, 2005-02-22.

[La05b] Lapkin, Anne: *Enterprise Architecture Research Agenda Set for 2005*. Gartner Research, ID Number: G00127039, 2005-04-14.

[LHM95] Lehner, Franz; Hildebrand, Knut; Maier, Ronald: *Wirtschaftsinformatik - Theoretische Grundlagen.* Carl Hanse Verlag, München, 1995, ISBN 3-446-18002-8.

[LMW05a] Lankes, Josef; Matthes, Florian; Wittenburg, André: *Softwarekartographie: Systematische Darstellung von Anwendungslandschaften*. In: 7. Internationale Tagung Wirtschaftsinformatik 2005, Bamberg, Germany, 2005.

[LMW05b] Lankes, Josef; Matthes, Florian; Wittenburg, André: *Management von Anwendunglandschaften: Adaption des IEEE 1471 für die Softwarekartograpie*. In: Software Engineering 2005, Essen, Germany, 2005.

[LMW05c] Lankes, Josef; Matthes, Florian; Wittenburg, André: *Softwarekartographie als Beitrag zum Architekturmanagement*. (noch nicht erschienen).

[MS03] McKeen, James D.; Smith, Heather A.: *Making IT happen – Critical Issues in IT Management*. Wiley Verlag (UK), West Sussex, 2003, ISBN 0470850876.

[Mü98] Müller, Wolfgang: *Metamodellierung als Instrument der Verknüpfung von Unternehmensmodellen*. Druckhaus Berlin-Mitte GmbH, Berlin, 1998, ISBN 3-8167-5164-4.

[MW04a] Matthes, Florian; Wittenburg, André: *Softwarekarten zur Visualisierung von Anwendungslandschaften und ihren Aspekten – Eine Bestandsaufnahme*. Technische Universität München, Fakultät für Informatik, Lehrstuhl für Informatik 19 (sebis), Technischer Bericht, 2004.

[MW04b] Matthes, Florian; Wittenburg, André: *Softwarekartographie: Visualisierung von Anwendungslandschaften und ihre Schnittstellen*. Technische Universität München, Fakultät für Informatik, Lehrstuhl für Informatik 19 (sebis), Technischer Bericht, 2004.

[NS03] Natis, Yefim V.; Schulte, Roy W.: *Introduction to Service-Oriented Architecture*. Gartner Research, ID Number: SPA-19-5971, 2003-04-14.

[Oe03] Oestereich, Bernd; Weiss, Christian; Schröder, Claudia; Weilkiens, Tim; Lenhard, Alexander: *Objektorientierte Geschäftsprozessmodellierung mit der UML*. dpunkt.verlag, Heidelberg, 2003.

[Oe04] Oestereich, Bernd: *Die UML 2.0 Kurzreferenz für die Praxis.* 3. Auflage, Oldenbourg Verlag, München, 2004, ISBN 3-486-27604-2.

[OGC05] The Office of Government Commerce (OGC): *Accountabilities*. <http://www.ogc.gov.uk/> (abgerufen am 2005-05-25).

[OMG03a] OMG: *Unified Modeling Language (UML) Specification: Infrastructure*. Version 2.0, Object Management Group, 2003.

[OMG03b] OMG: *Meta Object Facility (MOF) 2.0 Core Specification.* Version 2.0, Object Management Group, 2003.

[OMG04] OMG: *IT Portfolio Management Facility (ITPMF) Specification*. Final Submission, Object Management Group, 2004.

[Re82] Reisig, Wolfgang: *Petrinetze – eine Einführung*. Springer-Verlag, Berlin, Heidelberg, 1982.

[Ro03] Ross, Jeanne W.: *Creating A Strategic IT Architecture Competency: Learning In Stages*. In: MIS Quarterly Executive 2 (1), University of Minnesota, 2003.

[Ro95] Robinson, Arthur; Morrison, Joel; Muehrcke, Phillip; Kimerling, Jon; Guptill, Stephen: *Elements of Cartographiy*. 6. Auflage, Wiley, 1995, ISBN 0471555797.

[RP99] Rechenberg, Peter; Pomberger, Gustav: *Informatik-Handbuch*. 2. Auflage, Carl Hanser Verlag, München, Wien, 1999, ISBN3-446-19601-3.

[RS02] Rautenstrauch, Claus; Schulze, Thomas: *Informatik für Wirtschaftswissenschaftler und Wirtschaftsinformatiker*. Springer Verlag, Berlin, 2002, ISBN 3540411550.

[Sc01] Scheer, August W.: *ARIS – Modellierungsmethoden, Metamodelle, Anwendungen*. 4. Auflage, Springer Verlag, Berlin, Heidelberg, 2001, ISBN 3-540-41601-3.

[Sc02] Scheer, August W.: *ARIS – Vom Geschäftsprozess zum Anwendungssystem*. 4. Auflage, Springer Verlag, Berlin, Heidelberg, 2002, ISBN 3-540-65823-8.

[Sc03] Schulman, Jeff: *Enterprise Architecture: Benefits and Justification*. Gartner Research, ID Number: LE-19-4935, 2003-03-11.

[Sc04a] Schulman, Jeff: *Patterns and Bricks Are an Architect's Two Best Friends*. Gartner Research, ID Number: COM-21-7390, 2004-01-05.

[Sc04b] Schulman, Jeff: *Things to Get Right Before Architecture Development*. Gartner Research, ID Number COM-21-7450, 2004-01-5.

[Sc94] Schneider, Dieter: *Allgemeine Betriebswirtschaftslehre*. Oldenbourg Verlag, München, Wien, 1094, ISBN 3486203770.

[Sc98] Schütte, Reinhard: *Grundsätze ordnungsmäßiger Referenzmodellierung – Konstruktion konfigurations- und anpassungsorientierter Modelle*. Gabler Verlag, 1998, ISBN 3409128433.

[Se05] Sekatzek, Peggy: *Visualisierung von IT-Bebauungsplänen in Form von Softwarekarten – Konzeption und prototypische Umsetzung*. Diplomarbeit, Technische Universität München, Fakultät für Informatik, 2005.

[Seb04] Sebis: *Softwarekartographie Wiki*. Technische Universität München, Fakultät für Informatik, Lehrstuhl für Informatik 19 (sebis); <http://wiki.softwarekartographie.de/ index.php/Hauptseite> (abgerufen am 2005-05-12).

[So01] Sommerville, Ian: *Software Engineering*. 6. Auflage, Pearson Studium, Addison Wesley, München, 2001, ISBN 3-8273-7001-9.

[St73] Stachowiak, Herbert: *Allgemeine Modelltheorie*. 1. Auflage, Springer Verlag, Wien, 1973, ISBN 3-211-81106-0.

[SZ92] Sowa, John; Zachman, John: *Extending and formalizing the framework for information systems architecture*. In: IBM Systems Journal 31 (3), 1992.

[Te99] Teubner, Rolf Alexander: *Organisations- und Informationssystemgestaltung*. Dt. Univ.-Verl, Wiesbaden, 1999.

[To04] Torre, Leendert.; Lankenhorst, M.; Doest, H., Campschroer, J; Arbab, F.: *Landscape Maps for Enterprise Architectures*. Information Centre of Telematica Instituut, Technischer Bericht TI/RS/2004/016, Niederlande, 2004.

[TOG05] The Open Group: *TOGAF - Frequently Asked Questions*. <http://www.opengroup.org/ architecture/togaf8-doc/arch/p1/togaf_faq.htm> (abgerufen 2005-05-16).

[WKW94] WKWI: *Profil der Wirtschaftsinformatik*. Ausführungen der Wissenschaftlichen Kommission der Wirtschaftsinformatik, In: Wirtschaftsinformatik 36 (1), 1994.

[Wö02] Wöhe, Günter: *Einführung in die Allgemeine Betriebswirtschaftslehre.* 21. Auflage, Verlag Franz Vahlen, München, 2002, ISBN 3-8006-2865-1.

[Za04] The Zachman Institute for Framework Advancement: *Mission Statement* und *Zachman Framework*. <http://www.zifa.com> (abgerufen 2004-12-14).

[Za87] Zachman, John: *A framework for information systems architecture*. In: IBM Systems Journal 26 (3), 1987.