Softwarekarten zur Visualisierung von Anwendungslandschaften und ihren Aspekten - Eine Bestandsaufnahme

Florian Matthes, André Wittenburg

Technischer Bericht des Lehrstuhls für
Software Engineering betrieblicher Informationssysteme
Ernst Denert-Stiftungslehrstuhl
Lehrstuhl für Informatik 19
Technische Universität München

Boltzmannstraße 3, 85748 D-Garching

{florian.matthes, andre.wittenburg}@in.tum.de

26. März 2004

Zusammenfassung

Die Langlebigkeit und die steigende Anzahl von Informationssystemen führt zu immer komplexeren Anwendungslandschaften, die von Unternehmen zunehmend als vernetztes System betrachtet werden, um Strategien zu verwirklichen und Projekte zu planen. In diesem Papier berichten wir über erste Ergebnisse einer Forschungsstudie in Zusammenarbeit mit deutschen Unternehmen, in der untersucht wird, wie *Softwarekarten*, also grafische Darstellungen der Anwendungslandschaft und der sie betreffenden Kennzahlen, für die genannten Managementaufgaben genutzt werden.

Wir stellen zunächst interessante Beispiele existierender Softwarekarten und der durch sie verfolgten Ziele vor. Ein weiteres Ergebnis der Studie sind Anforderungen, die von Unternehmen an Softwarekarten und an Werkzeuge zu ihrer Bearbeitung und Nutzung gestellt werden.

Das langfristige Ziel unserer Forschung ist es, einen grundlegenden Begriffsapparat und anschauliche Notationen für Softwarekarten zu entwickeln, um komplexe betriebliche Informations-Infrastrukturen zu beschreiben, zu bewerten und langfristig zu gestalten.

1 Einleitung

Betriebliche Informationssysteme nehmen in heutigen Unternehmen eine bedeutende Rolle ein, wichtige Prozesse der Wertschöpfungskette werden von ihnen unterstützt bzw. laufen automatisiert ab.

Diese Bedeutung betrieblicher Informationssysteme ist seit Bestehen dieser stets gestiegen, der Ausfall oder die Fehlfunktion eines oder mehrerer Informationssysteme kann die Handlungsfähigkeit eines Unternehmens stark einschränken. Eine Bank, deren zentrales Buchungssystem nicht verfügbar ist, ein Versandhaus, dessen Logistiksystem seinen Dienst versagt, oder ein Automobilhersteller, dessen Produktionsplanungssystem ausfällt, sind exemplarische Szenarien, die zu einer Einschränkung der Handlungsfähigkeit des betroffenen Unternehmens führt und die Bedeutung von Informationssystemen untermauert. Die volle Einsatzfähigkeit eines Unternehmens hängt jedoch nicht an einem einzelnen Informationssystem, sondern an der Gesamtheit, die wir als Anwendungslandschaft bezeichnen.

Eben diese Anwendungslandschaften zeichnen sich in großen Unternehmen durch zunehmende Komplexität aus, die sich u.a. durch eine steigende Anzahl von Informationssystemen und ihre hohe Lebensdauer ergibt. Ziel unseres Forschungsprojektes ist es, ein geeignetes Modell zur Beschreibung von Anwendungslandschaften zu entwickeln und die verschiedenen Anforderungen von Unternehmen auf eine möglichst intuitive Art und Weise in diesem Modell umzusetzen. Zusätzlich soll die Verträglichkeit mit existierenden Modellen der Informatik gewährleistet sein.

In einem ersten Schritt ist insbesondere eine eindeutige Begriffswelt von Bedeutung, da im Bereich Anwendungslandschaften verschiedene Synonyme und Homonyme verwendet werden, zu diesem Zweck definieren wir in Abschnitt 2 einige Begriffe. Eine kurze Vorstellung unseres Forschungsprojektes *Softwarekartographie* und eine Verknüpfung der relevanten Domänen erfolgt ebenso in diesem Abschnitt.

Durch die Befragung unserer Projektpartner, große deutsche Unternehmen verschiedener Branchen, hat sich die These der Komplexität von Anwendungslandschaften manifestiert. Die Anwendungslandschaft der befragten Unternehmen besteht je nach Branche aus über 1.000 Informationssystemen. Einige der befragten Unternehmen haben bereits Anstrengungen unternommen, eine geeignete Darstellung zur Beschreibung ihrer Anwendungslandschaft zu erhalten. Einige dieser Ansätze zur Beschreibung sind in Abschnitt 3 wiedergegeben.

Durch die Analyse der existierenden Ansätze und die Befragung der Unternehmen ergeben sich Anforderungen an ein Modell zur Beschreibung einer Anwendungslandschaft. Die ersten Ergebnisse dieser Anforderungsanalyse präsentieren wir in Abschnitt 4.

Existierende Modelle der Informatik und Wirtschaftswissenschaften zur Beschreibung von Softwaresystemen (z.B. UML [FS99]) und für Geschäftsprozesse (z.B. Ereignisgesteuerte Prozessketten [Sch95]) betrachten wir im Kontext von Anwendungslandschaften in Abschnitt 5. Ebenso werden Werkzeuge von Softwareherstellern vorgestellt, die entweder einen direkten Bezug zu Anwendungslandschaften besitzen oder durch ihre Modellierungsfähigkeit zu dessen Beschreibung eingesetzt werden können.

Abschließend folgt in Abschnitt 6 eine Dikussion der bisherigen Ergebnisse mit einer Zusammenfassung.

2 Analogie, Betrachtungsebenen und Domänen

Zur Verdeutlichung der Motivation für das Forschungsprojekt *Softwarekartographie* verwenden wir folgende Analogie:

Analog zur Stadt- oder Landschaftsplanung liegt der Fokus bei der Gestaltung und Planung von

komplexen betrieblichen Informations-Infrastrukturen auf einer langfristigen, investitionssichernden Perspektive. Die Langlebigkeit von Gebäuden, die beschränkte Ressource "Raum" und die komplexen Wechselwirkungen zwischen individuell entwickelten Gebäuden (Verkehr, Lärm, Licht, Brandschutz etc.) führen zu übergeordneten Planungs- und Gestaltungsprozessen (Quartiersplanung, Städteplanung, Landschaftsplanung etc.) gegenüber denen eines Architekten, der den Fokus auf einzelne Gebäuden legt. Diese Abstraktionsstufe vom Stadt- oder Landschaftsplaner zum Architekten ist vergleichbar mit der eines CIOs oder IT-Managers, der die Planung und Gestaltung der gesamten Anwendungslandschaft übernimmt, gegenüber dem eines Softwarearchitekten, der einzelne Softwaresysteme entwickelt.

Die verschiedenen Sichten auf die Anwendungslandschaft sind nicht getrieben durch die Sichten von Herstellern für Standard- oder Individualsoftware, sondern durch die Unternehmen, ihre Organisationseinheiten und ihre Nutzer. Unternehmen benötigen diese Sichten, um beispielsweise Programme und Projekte zu planen, die Veränderungen ihrer Anwendungslandschaft zu erkennen, strategische Ziele zu verfolgen und trotz der hohen Komplexität ihre Anwendungslandschaft zu beherrschen.

2.1 Betrachtungsebenen

Die Softwarekartographie – mit ihren Modellen und Methoden – soll die Darstellung der gesamten Anwendungslandschaft ermöglichen und die verschiedenen Betrachtungsebenen in Abbildung 1 verbinden.

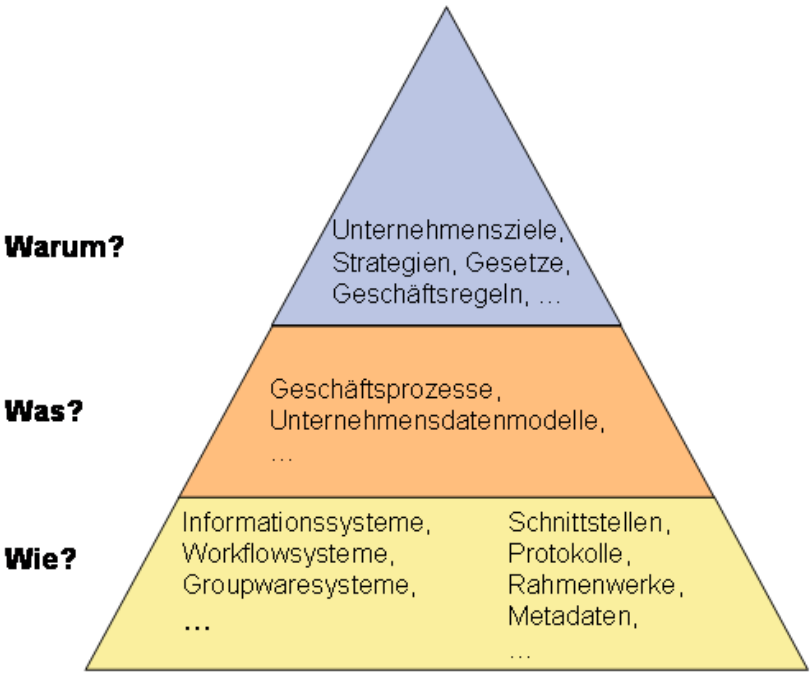

Abbildung 1: Betrachtungsebenen der Softwarekartographie

Die unterste Betrachtungsebene beschäftigt sich mit den typischen Aspekten und Kennzahlen von Informationssystemen. Schnittstellen-Beziehungen und genutzte Protokolle sind für die System-Interkommunikation relevant, Groupware-, Workflow- und allgemeine Middleware-Systeme unterstützen die einzelnen Informationssysteme bei der Aufgabenerfüllung. Die Frage "Wie tue ich

es?" ist die Verallgemeinerung dieser Betrachtungsebene; die Artefakte dieser Ebene werden auf der nächst höherliegende Ebene "Was?" implementiert.

Geschäftsprozesse, Unternehmensdatenmodelle etc. repräsentieren bzw. stellen das Kerngeschäft eines Unternehmens in abstrakter Form dar. Die Wertschöpfungskette, als zumeist abstraktestes Medium zur Beschreibung des Kerngeschäftes, symbolisiert das Produktionsziel eines Unternehmens. Die Frage "Was tue ich?" beschreibt diese Ebene und verbindet die Implementierungsebene "Wie?" mit der Ziel/Strategie-Ebene "Warum?".

Unternehmensziele, Strategien ebenso wie gesetzliche Regelungen haben einen indirekten oder direkten Einfluss auf die Gestaltung, Funktionsweise und Aufgabe von Informationssystemen. Ein strategisches Ziel, das die Vorgabe macht, die Anzahl von Individualsoftware zu reduzieren, hat direkten Einfluss auf die Anwendungslandschaft, gesetzliche Regelungen wiederum (z.B. MaK oder Basel II im Bereich Banken) verändern Informationssysteme indirekt, da zunächst die Geschäftsprozesse und Regeln betroffen sind, diese jedoch in den Informationssystemen abgebildet werden müssen. Dem Einfluss dieser Ziele, Strategien etc. auf die Anwendungslandschaft wird auf der Betrachtungsebene "Warum tut ich es?" Rechnung getragen.

Die einzelnen Betrachtungsebenen im Kontext der Anwendungslandschaft zu verbinden und zu einer intuitiven Beschreibung und Darstellung zu gelangen, ist eine Aufgabe der *Softwarekartographie*.

2.2 Begriffsdefinitionen

Im Rahmen der Anforderungsanalyse mit unseren Projektpartnern wurden verschiedene Begriffe im Kontext des Forschungsprojektes verwendet. Die folgende Liste fasst die wichtigsten Begriffsdefinitionen für die Softwarekartographie zusammen:

Anwendungslandschaft: Die Gesamtheit aller betrieblichen Informationssysteme in einem Unternehmen.

Softwarekarte: Eine graphische Repräsentation der Anwendungslandschaft oder Ausschnitte selbiger. Eine Softwarekarte setzt sich zusammen aus einer oder mehrerer Sichten, die verschiedene Aspekte visualisieren.

Aspekt: Beschreibt eine oder mehrere Eigenschaften von Informationssystemen.

Kennzahl: Erfasst Aspekte von Informationssystemen quantitativ und in konzentrierter Form.

Sicht: Spezielle Ausprägung einer Softwarekarte, auf der bestimmte Aspekte dargestellt werden. Verschiedene Sichten können kombiniert werden, um andere Softwarekarten zu erzeugen.

Softwarekartographie: Beschreibt die Modelle und Methoden zur Beschreibung und graphischen Darstellung von Anwendungslandschaften durch Softwarekarten.

Für den Begriff *Anwendungslandschaft* wurde von unseren Projektpartnern auch der Begriff *Systemlandschaft* verwendet. Dieser Begriff wird oft im Kontext von technischen Infrastrukturen bzw. *Hardwarelandschaften* verwendet und bezeichnet dort die verwendeten Hardwaresysteme, so dass dieser Begriff missverständlich ist und wir diesen Begriff nicht für Informationssystem-Infrastrukturen verwenden.

Der Begriff *Kennzahl* wird innerhalb diesen Kontextes verwendet, um quantitative Eigenschaften von Informationssystemen wie beispielsweise "Anzahl von Nutzern" zu beschreiben. Da aber auch nicht quantitative Eigenschaften, wie "Geschäftsprozesse" oder "Organisationseinheiten" relevant sind, wurde der abstraktere Begriff *Aspekt* eingeführt, so dass Kennzahlen eine Untermenge von Aspekten bilden.

2.3 Domänen der Softwarekartographie

Die Softwarekartographie als Disziplin zur Beschreibung, Bewertung und Gestaltung von komplexen betrieblichen Informations-Infrastrukturen grenzt an verschiedene existierende Wissenschaften und andere Domänen an. Abbildung 2 visualisiert die einzelnen für die Softwarekartographie relevanten Domänen, die bei der Entwicklung der Modelle und Notationen eine Rolle spielen.

Die Informatik, die Wirtschaftswissenschaften und die Kartographie sind die Eckpfeiler der Softwarekartographie. Bei der Informatik ist insbesondere der Bereich des Software-Engineering mit seinen Modellen und Methoden (Softwarearchitekturen, Vorgehensmodelle, UML etc.) von Bedeutung, die Wirtschaftswissenschaften liefern für die Softwarekartographie neben dem Prozessmanagement (Prozesse, Organisationen etc.) ebenso die Kostenaspekte, die für die Softwarekartographie relevant sind.

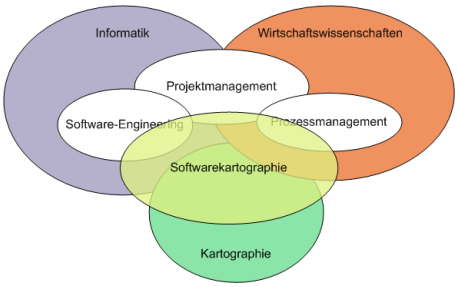

Abbildung 2: Domänen der Softwarekartographie

Das Projektmanagement als Querschnitts-Domäne steuert und plant den steten Wandel der Anwendungslandschaft und resultiert in zeitabhängigen Darstellungen, die neben Ist-Anwendungslandschaften auch Plan- bzw. Soll-Anwendungslandschaften repräsentieren.

Die Kartographie

> (Cartography –) the discipline dealing with the conception, production, dissemination and study of maps [HGM02]

soll mit ihrem Verständnis zur Erstellung von Karten, der Nutzung von Farben, Formen etc. helfen Softwarekarten zu entwickeln, die intuitiv und anwendungsbezogen *korrekt* sind.

3 Ansätze zur Beschreibung von Anwendungslandschaften

Der existierende Handlungsbedarf bei den Projektpartnern, einen Überblick und ein Management-Instrument für die Anwendungslandschaft zu erhalten, hat zu unterschiedlichen Ansätzen geführt, die Anwendungslandschaft zu erfassen und zu visualisieren.

Im Folgenden wollen wir einige Ansätze vorstellen, die unsere Projektpartner entwickelt haben, um ihre primären Anforderungen zu erfüllen. Diese und weitere Anforderungen, die wir während unserer Arbeit mit den Projektpartnern aufgenommen haben, werden in Abschnitt 4 vorgestellt.

Die einzelnen Beispiele in den folgenden Unterabschnitten entsprechen vom Aufbau den Softwarekarten einzelner Projektpartner, die eine Softwarekarte entwickelt haben. Die Beispiele sind jedoch hinsichtlich der tatsächlichen Systeme[1] anonymisiert und die Komplexität stark reduziert, da ausschließlich die Syntax und Semantik vorgestellt werden soll. Die Originale dieser Softwarekarten zeigen über 100 Informationssysteme und werden in gedruckter Form im DIN A1 oder DIN A0 Format verwendet.

[1] Wird im Folgenden von Systemen gesprochen, so sind Informationssysteme gemeint, die durch eine oder mehrere Applikationen implementiert sind.

3.1 Softwarekarte A

Ziel der Softwarekarte A ist die Visualisierung aller Systeme des Unternehmens, die Zuordnung der Systeme zu Funktionsbereichen und die Darstellung der Schnittstellen-Beziehungen zwischen diesen Systemen (Beispiel in Abbildung 3). Diese Art einer Softwarekarte ist von einem Unternehmen aus der Versicherungsbranche entwickelt worden.

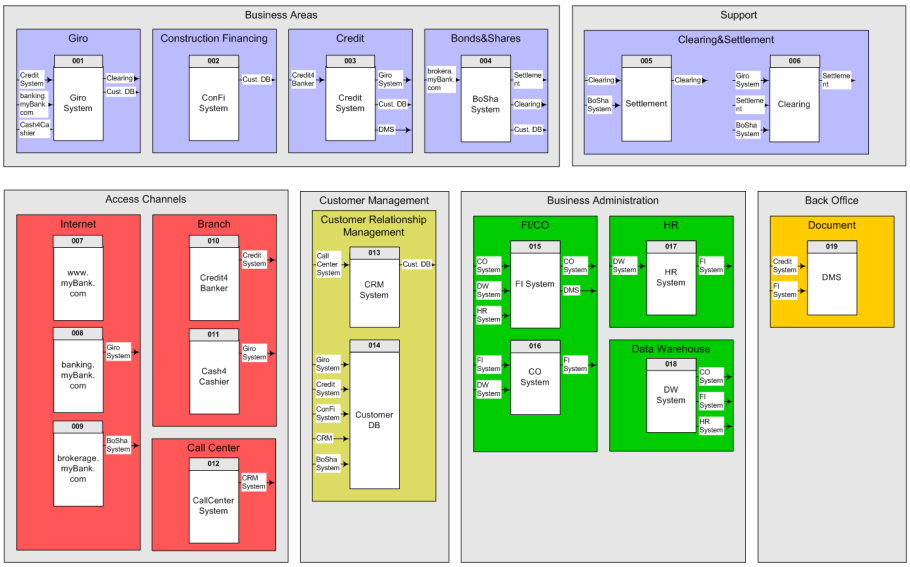

Abbildung 3: Beispiel für Softwarekarte A

Die gröbste Anordnung der Elemente auf der Karte erfolgt anhand einer logischen Gruppierung (*Business Areas*, *Access Channels* etc.), die individuell für das Unternehmen zutrifft und durch Rechtecke mit einem grauen Hintergrund visualisiert werden. Diese Gruppierung wird durch eine weitere Untergruppierung verfeinert, innerhalb der Sparten kann dies bei einer Versicherung beispielsweise Leben, Rechtsschutz, Kraftfahrt etc. sein, bei einer Bank wären Kredit, Spar, Giro etc. denkbar.

Die farbliche Unterscheidung in dieser Softwarekarte ist getrieben durch einen *Farbcode*, der den einzelnen Gruppen, die durch Rechtecke dargestellt werden, bestimmte Farben zuweist. Dieser *Farbcode* kann beispielsweise, ebenso wie die logische Gruppierung in Funktionsbereiche, unternehmensweit genutzt werden, um Bereiche farblich hervorzuheben.

Systeme werden den Untergruppen zugeordnet, wobei ein System in mehreren Untergruppen vertreten sein darf, wenn eine Mehrfachnennung zu der Gruppierung passt. Ein System wird durch ein Rechteck mit einer abgehobenen Kopfzeile dargestellt, wobei in der abgesetzten Kopfzeile eine eindeutige Systemnummer angegeben ist und die Beschriftung des Rechtecks den Namen des Systems enthält.

Kommunizieren zwei Systeme über eine Schnittstelle miteinander, so wird dies mit zwei Pfeilen visualisiert. Der Server erhält einen eingehenden Pfeil, der mit dem Namen des Clients beschriftet ist; der Client entsprechend einen ausgehenden Pfeil, beschriftet mit dem Namen des Servers.

Die Elemente auf der Softwarekarte A erfahren keine spezifische Anordnung der Elemente entlang der X- oder Y-Achse. Die Positionierung der Gruppen auf der Softwarekarte, der Untergruppen innerhalb der Gruppen und der Systeme in den Untergruppen wird platzoptimierend vorgenommen.

3.2 Softwarekarte B

Analog zu Softwarekarte A soll Softwarekarte B alle Systeme des Unternehmens und die Schnittstellen-Beziehungen zwischen den Systemen visualisieren (Beispiel in Abbildung 5). Im Gegensatz zu Softwarekarte A sind jedoch die Linien zur Darstellung von Schnittstellen-Beziehungen nicht nur durch Pfeile angedeutet, sondern verbinden die Systeme. Diese Art einer Softwarekarte wird ebenso von einem Versicherungskonzern genutzt.

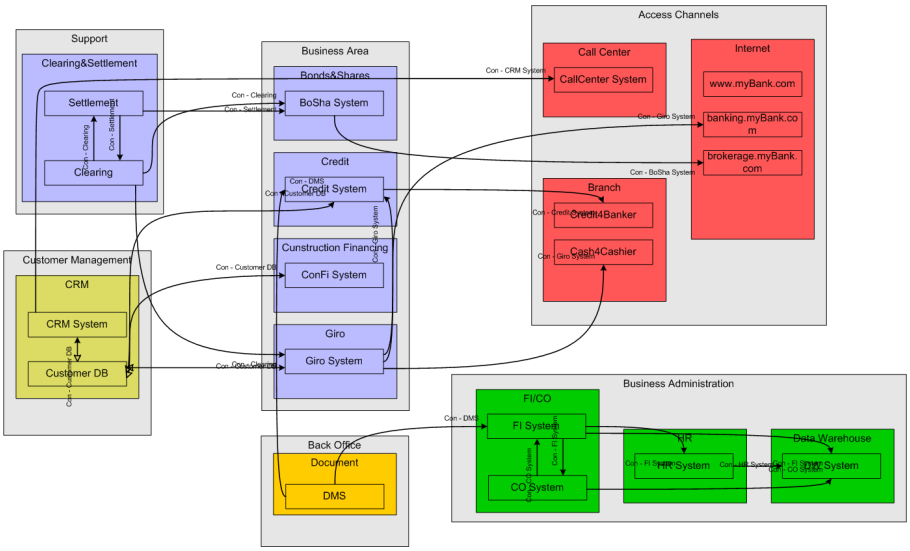

Abbildung 4: Beispiel für Softwarekarte B

Die Elemente auf der Softwarekarte sind, ebenso wie die Zielsetzung, vergleichbar mit der von Softwarekarte A. Der Einsatz von Farben dient hier ebenso zum Hervorheben von logischen Gruppierungen; bei Schnittstellen-Beziehungen wird zusätzlich durch unterschiedliche Pfeilarten zwischen "lesenden" und "schreibenden" Schnittstellen unterschieden.

Die Übersichtlichkeit dieser Karte leidet, wenn eine entsprechende Verzahnung der Systeme über Schnittstellen gegeben ist. Auch zentrale Komponenten wie ein Portal oder eine EAI-Komponente[2] würden durch ihre zahlreichen Schnittstellen-Beziehungen die Übersichtlichkeit stark mindern. Jedoch zeigt eine derartige Darstellung die Komplexität der Verzahnung der Anwendungslandschaft deutlicher als Softwarekarte A.

3.3 Softwarekarte C

Die Softwarekarte C soll zum einen eine Transparenz der Anwendungslandschaft schaffen und zum anderen IT-Projekte mit den betroffenen Systemen und deren Entwicklungsstand bzw. Projektfortschritt visualisieren (Beispiel in Abbildung 4).

Im Gegensatz zu den Softwarekarte A und B wird bei Softwarekarte C eine Anordnung entlang der X- und Y-Achsen vorgenommen. Die X-Achse enthält die Prozessschritte der Wertschöpfungskette des Unternehmens, die Y-Achse unterscheidet die Systeme nach Kriterien wie beispielsweise dispositiv und operativ. Die Prozessschritte auf der X-Achse sind durch Chevrons dargestellt, die Unterscheidung auf der Y-Achse erfolgt fließend durch ein Rechteck mit mehreren Beschriftungen. Diese Art einer Softwarekarte wurde von einem Service-orientierten Logistik-Konzern erstellt.

[2]EAI - Enterprise Application Integration

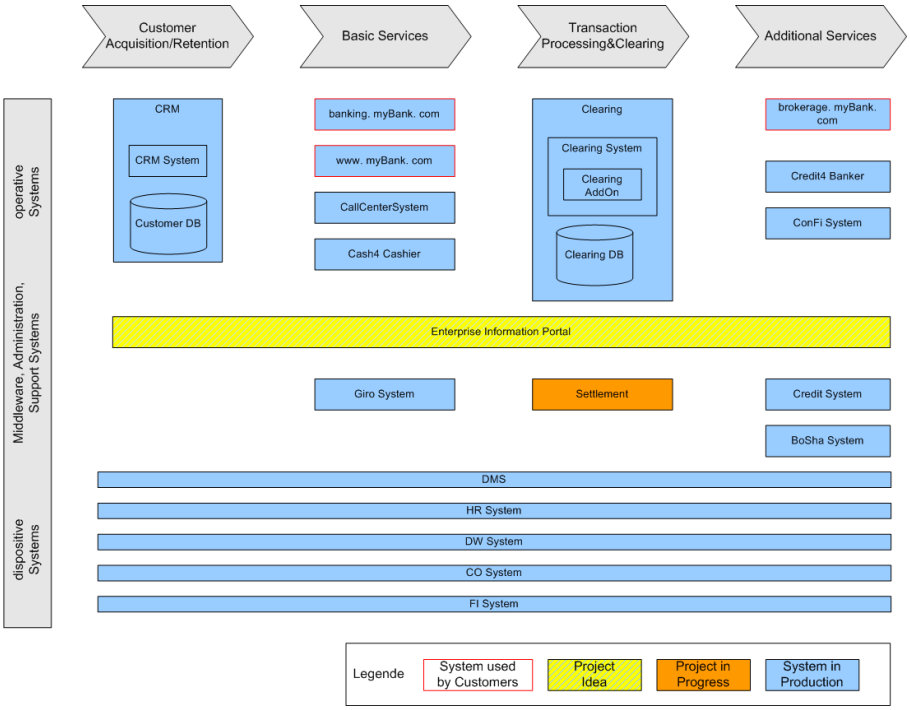

Abbildung 5: Beispiel für Softwarekarte C

Die Elemente auf der Softwarekarte wurden von dem betreffenden Unternehmen wie folgt definiert:

System: Ein System ist eine logische Zusammenfassung physisch voneinander abhängiger Komponenten, die mittels Funktion und Daten Geschäftsprozesse oder Teile davon unterstützen. Zu einem System gehört eine integrierte Datenhaltung.

Applikation: Eine Applikation ist die zusammenhängende und funktionsfähige Schnittstelle gegenüber einem Anwender.

Teilapplikation: Mehrere Teilapplikation zusammen bilden eine Applikation.

Datenbank: Eine Datenbank ist eine Sammlung von Daten, die zueinander in Beziehung stehen, systematisch oder methodisch angeordnet und einzeln mit Hilfe elektronischer Mittel zugänglich sind.[3]

Die ersten drei Elemente (System, Applikation und Teilapplikation) werden je durch ein Rechteck dargestellt. Eine Datenbank wird durch das typische Datentopfsymbol visualisiert. Die Beschriftung dieser vier Elemente erfolgt mit einem Kurznamen und einer Kurzbeschreibung[4]. Zusätzlich gilt: Wenn der Systemname gleich dem Namen der Applikation und dem Namen der Datenbank ist, so wird nur ein Rechteck für das System dargestellt.

Zur Darstellung

- verschiedener Status der Systeme, wie beispielsweise “System in Betrieb” oder “System wird geändert” und

[3]Eine Unterscheidung zwischen Datenbank (DB), Datenbanksystem (DBS) und Datenbankmanagementsystem (DBMS) entfällt hier. Der hier verwendete Begriff *Datenbank* würde nach [Dat00] einer Datenbank in Kombination mit einem Datenbanksystem und einem Datenbankmanagementsystem entsprechen.

[4]In der exemplarischen Darstellung in Abbildung 4 wird auf die Kurzbeschreibung verzichtet.

- weiterer Eigenschaften der Systeme, wie “System wird von Kunden benutzt” oder “System ohne Benutzerschnittstelle”,

werden verschiedene Farben und Schraffuren für Hintergründe und Farben für Rahmen der Elemente gewählt. In dem Beispiel in Abbildung 4 bedeutet ein roter Rahmen, dass das betreffende System direkt von Kunden bedient wird.

Die Darstellung von IT-Projekten wird durch die Wahl einer geeigneten Hintergrundfarbe des Elementes erreicht, wobei das Rechteck kein real existierendes System darstellen muss, sondern lediglich ein IT-Projekt, welches zu einem System führen kann, jedoch nicht muss. In Abbildung 4 ist ein “Enterprise Information Portal” entsprechend als IT-Projekt dargestellt.

3.4 Softwarekarte D

Die Zielsetzung der Softwarekarte D ist die Darstellung der genutzten Systeme in einzelnen (Teil-) Prozessschritten in Zusammenhang mit den Nutzern des Systems. Durch die gewachsene Anwendungslandschaft des Unternehmens werden in gleichen Prozessschritten unterschiedliche Systeme eingesetzt, so dass eine Darstellung benötigt wurde, die den Zusammenhang zwischen Prozessschritt und verwendeten System zeigt, um beispielsweise Optimierungspotential und Redundanzen zu erkennen. Ersteller dieser Art einer Softwarekarte ist ein Automobilhersteller.

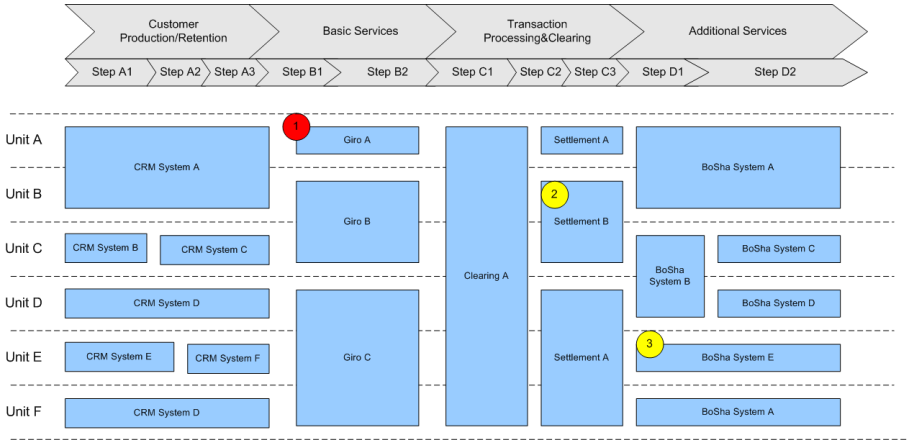

Abbildung 6: Beispiel für Softwarekarte D

Die Elemente auf der X-Achse sind wie bei Softwarekarte B an Hand des Geschäftsprozesses angeordnet, die Y-Achse gliedert nach Nutzern bzw. Nutzergruppen.

Ein Rechteck in der Softwarekarte symbolisiert ein Softwaresystem, welches mit seiner Systembezeichnung beschriftet ist. Eine horizontale Ausdehnung kennzeichnet, dass ein System in mehreren (Teil-)Prozessen verwendet wird, wobei die (Teil-)Prozesse im oberen Teil mittels ineinander übergehender Chevrons gekennzeichnet sind. Eine vertikale Ausdehnung bedeutet, dass mehrere Nutzer dasselbe System für diesen (Teil-)Prozess verwenden.

Die farbigen Kreise an einzelnen Systemen zeigen aktive oder geplante Projekte, die Nummer innerhalb des Kreises referenziert das entsprechende Projekt. Die Kreise werden zusätzlich genutzt, um beispielsweise einen Handlungsbedarf zu visualisieren, der durch schlechte Performanz oder Stabilität des Systems hervorgerufen wird.

3.5 Gemeinsamkeiten der Softwarekarten

Allen Softwarekarten ist gemein, dass die Größe[5] eines Symbols für ein System, Modul etc. keine Aussage über deren Bedeutung macht. In Softwarekarte A wird die Größe so gewählt, dass die einzelnen Schnittstellen-Beziehungen alle gleichmäßig angeordnet werden können; Softwarekarte D nutzt die Größe um die Zugehörigkeit zu mehreren Prozessschritten in der Wertschöpfungskette darzustellen. Keine der Darstellungen hinterlegt beispielsweise die Anzahl von Benutzern, die Anzahl von *Function Points* [Bal00] oder *Lines of Code* mittels der Größe einzelner Elemente.

Ebenso werden Farben so gewählt, dass sie deutlich unterschieden werden können, jedoch nicht um mittels Farbnuancen oder besonderer Farbschemata Zusammengehörigkeit bzw. Verwandheit in der Darstellung zu visualisieren. Bei der Darstellung von Kennzahlen (z.B. Wartungskosten) könnten Farbnuancen zur Kodierung verwendet werden, um die Aspekte geeignet zu repräsentieren.

Abstrahiert man von den obigen Unterschieden im Detail, so erkennt man, dass alle vorgestellten Ansätze für Softwarekarten das primäre Ziel haben, die Systeme eines Unternehmens nach Aufgabenbereichen, Geschäftsprozessen oder Nutzern zu ordnen und darzustellen, wobei die Softwarekarten A und B zusätzlich die Schnittstellen-Beziehungen visualisieren. Nur die Softwarekarten C und D visualisieren zusätzlich IT-Projekte mit Wirkung auf die Anwendungslandschaft. Keine der Darstellungen visualisiert Kosten, Nutzen, Transaktionsraten, Anzahl von Nutzern oder andere entscheidungsrelevante Aspekte, um Entscheidern ein Werkzeug zur weiteren Gestaltung der Anwendungslandschaft an die Hand zu geben.

Diese Tatsache resultiert aus dem derzeitigen Vorgehen bei der Erstellung der Softwarekarten und deren Pflege. Mit einer Ausnahme wurden alle Softwarekarten ausschließlich mittels Microsoft Visio, Microsoft PowerPoint erstellt, ohne dass Teile oder die ganze Karte generiert wurden. Einzelne Projektpartner nutzen teils mehrere Sichten, die jedoch getrennt voneinander gepflegt werden, so dass der Wartungs- und Pflegeaufwand sich weiter erhöht.

4 Anforderungen und relevante Aspekte für Softwarekarten

Die in Abschnitt 3 vorgestellten Ansätze zur Beschreibung von Anwendungslandschaften erfüllen unterschiedliche Anforderungen der betreffenden Unternehmen. Neben den bereits erfüllten Anforderungen, durch die jeweils eigene Beschreibung, haben wir in Zusammenarbeit mit den Projektpartnern weitere Anforderungen an Softwarekarten erhoben. Diese Anforderungen sind in Anhang A gelistet und lassen sich wie folgt gruppieren:

- Planerische Aspekte
- Fachliche Aspekte
- Technische Aspekte
- Wirtschaftliche Aspekte
- Operative Aspekte

Die *planerischen Aspekte* beinhalten Anforderungen, die sich durch die Veränderung der Anwendungslandschaft über die Zeit ergeben. Projekte und Programme[6], die Auswirkungen auf die Anwendungslandschaft haben, sollen auf einer Softwarekarte visualisiert werden, um beispielsweise Abhängigkeiten zwischen den Projekten zu erkennen.

[5] Die Größe wird hier mit der Fläche eines Symbols gleichgesetzt.

[6] Programm sei an dieser Stelle als Zusammenfassung mehrerer Projekte zu einem Programm zu verstehen.

Drei mögliche Einsatzszenarien, die durch eine Softwarekarte mit planerischen Aspekten unterstützt werden können, sind:

1. Ein Projekt hat das Ziel, ein System zu verändern oder abzulösen. Um die "Umgebung" des Projektes in der Anwendungslandschaft zu erfassen, muss ersichtlich sein, welche anderen Projekte und Systeme von diesem Projekt betroffen sind. Eng verbunden mit dieser Anforderung sind die technische Aspekte "Schnittstellen-Beziehungen" und "Schnittstellen-Konnektoren und Schnittstellen-Beziehungen" (siehe Anhang A), da insbesondere die Beziehungen zwischen Systemen für die Planung relevant sind. Projektleiter und Planer hätten durch eine Softwarekarte, die Projekte und Schnittstellen visualisiert, die Möglichkeit Abstimmungs-/Kommunikationsbedarf zu erkennen und entsprechend zu reagieren.

2. Zu Beginn einer Planungsperiode muss das vorhandene/genehmigte IT-Budget den beantragten Projekten zugeordnet werden. Nach einer Priorisierung und Auflösung von Abhängigkeiten zwischen den Projekten entsteht eine neue Planung, die die zurückgestellten bzw. zurückgezogenen Projekte nicht mehr enthält. Die Planer und Strategen erhalten durch eine derartige Softwarekarte die Möglichkeit, die ursprüngliche Planung der Anwendungslandschaft mit der neuen Planung zu vergleichen und ebenso über mehrere Planungsperioden die Entwicklung der Anwendungslandschaft zu überwachen.

3. Ein Unternehmen hat das strategische Ziel, die Zahl der Individual-Softwaresysteme zu reduzieren und diese durch Standard-Softwaresysteme zu ersetzen. Ein derartiger Vorgang kann sich über mehrere Planungsperioden erstrecken, so dass die zeitliche Veränderung über die einzelnen Planungsperioden in einer Softwarekarte angezeigt werden soll. Projektleiter, Strategen und Manager können durch eine derartige Softwarekarte erkennen, welche Systeme zu welchem Zeitpunkt abgelöst werden sollen und in welchen Perioden Veränderungen geplant sind. Eine Variation dieses Szenarios ist die Einführung eines *Enterprise Information Portals*, bei der verschiedene Systeme zeitversetzt an das Portal angeschlossen bzw. durch das Portal abgelöst werden.

Fachliche Aspekte lassen sich unterteilen in organisatorische und prozessorientierte Aspekte, die eng miteinander verzahnt sind, da Organisationseinheiten für die Durchführung bestimmter Prozessschritte verantwortlich sind und vice versa Prozessschritte von Personen, die Organisationseinheiten zugeordnet sind, durchgeführt werden.

Die Softwarekarten A bis D (siehe Abschnitt 3) sind Beispiele für Softwarekarten, die fachliche Aspekte enthalten. Die Softwarekarten A und B haben den Fokus auf den Organisationseinheiten bzw. Funktionsbereichen und gruppieren die einzelnen Systeme entsprechend dieses Aspektes; die Softwarekarten C und D ordnen die einzelnen Systeme entlang einer horizontalen Prozesskette.

Weitere fachliche Aspekt sind die Einordnung von Systemen in "operative Systeme", "dispositive Systeme" etc. (Softwarekarte C) und die Zuordnung von Nutzergruppen zu Systemen (Softwarekarte D).

Technische Aspekte erstrecken sich von der Implementierungssprache eines Systems, über die Schnittstellen bis hinzu Eigenschaften wie Architektur oder genutzter Middleware. Diesen technischen Aspekten wird zum Teil in existierenden Visualisierungssprachen Rechnung getragen. UML kann beispielsweise mit den Verteilungs- und Komponentendiagramm die Architektur eines Informationssystems modellieren, Schnittstellen und genutzte Middleware können ebenso abgebildet werden.

Die Softwarekartographie adressiert diese Anforderungen auf der Ebene der Anwendungslandschaft, so dass nicht die Schnittstellen-Beziehungen innerhalb eines Informationssystems von Relevanz sind, sondern zwischen allen Informationssystemen. Durch die Verschiebung des Fokus auf die Anwendungslandschaft lassen sich insbesondere transitive Abhängigkeiten erkennen und Projekte planen (siehe auch planerische Aspekte).

Durch den Bezug der Softwarekartographie zur Anwendungslandschaft können existierende Aspekte anders genutzt werden. Die Programmiersprachen der Informationssysteme in Kombination mit der Eigenschaft Standard-/Individualsoftware gibt Aufschluss über das benötigte Know-how innerhalb eines Unternehmens. Durch den Wandel von modularen Programmiersprachen zu objektorientierten können in Unternehmen viele Mitarbeiter mit Pascal oder Modula Know-how arbeiten, jedoch der prozentuale Anteil an Systemen mit objektorientierten Programmiersprachen stark gestiegen sein, so dass ggf. Kapazitätsüberschüsse bzw. -engpässe entstehen, die so erkannt werden können.

Die *wirtschaftlichen Aspekte* umfassen die verschiedenen Kostenarten, die bei der Entwicklung, dem Betrieb, der Wartung, dem Einkauf, etc. von Informationssystemen entstanden sind, entstehen bzw. entstehen werden. Die verschiedenen Kostenarten sollen miteinander kombiniert werden und mit der Anwendungslandschaft in Verbindung gebracht werden, so dass die Softwarekartographie als Hilfsmittel beim IT-Controlling dienen kann.

Die Betrachtung von Bereitschaftskosten in Kombination mit Wartungskosten per anno, kann Systeme aufzeigen, die im Vergleich zu anderen Systemen einen höheren Wartungsaufwand haben, jedoch im Betrieb vergleichsweise günstig sind.

Die Möglichkeit Kosten-Kennzahlen bei der Darstellung von Anwendungslandschaften zu berücksichtigen, wurde bisher von den Projektpartnern nur bedingt wahrgenommen. Welche Metriken und Kennzahlen jedoch für das IT-Controlling in diesem Kontext sinnvoll und nutzbar sind, muss sich im Rahmen des weiteren Vorgehens zeigen. Insbesondere die Problematik, dass nicht alle Unternehmen eine Zuordnung zwischen Kosten und Informationssystemen haben, erschwert den Aufbau von derartigen Kennzahlensystemen.

Operative Aspekte beziehen sich auf den unmittelbaren Betrieb von Informationssystemen und die verbundenen Ereignisse. Der Aspekt *Betriebsort* eines Informationssystems gibt beispielsweise die geographische Position des bzw. der physikalischen Server(s) eines Informationssystems an. Dieser Betriebsort kann von dem/den Nutzungsort(en) eines Informationssystems abweichen, wobei die Information "Betriebs- vs. Nutzungsort" von Projektpartnern als wertvoll erachtet wird.

Ein anderer operativer Aspekt ist der Ablauf von verschiedenen *Batch-Programmen*, die sich in gegenseitiger Abhängigkeit befinden und zeitlich versetzt laufen müssen. Das Visualisieren derartiger Abhängigkeiten und Abläufe in Anwendungslandschaften unter Nutzung von existierenden Diagrammen (z.B. Gantt), wird bereits von Projektpartnern durchgeführt.

Kombiniert man die Gruppen von relevanten Aspekten mit den Betrachtungsebenen (siehe Abschnitt 2.1), so ergibt sich folgendes Bild:

Planerische Aspekte stellen eine Querschnittsaufgabe dar und ziehen sich durch alle Betrachtungsebenen. Strategien auf der Betrachtungsebene "Warum?" haben ebenso einen planerischen Anteil, wie Änderungen in Geschäftsprozessen (Betrachtungsebene "Was?") oder die Veränderung von Schnittstellen (Betrachtungsebenen "Wie?").

Fachliche Aspekte finden sich in den Betrachtungsebenen "Warum?" und "Wie?". Organisationseinheiten sind durch die Aufbauorganisation getrieben und schlagen sich in den Geschäftsprozessen nieder.

Technische Aspekte sind in der Majorität der Betrachtungsebene "Wie?" zuzuordnen, Programmiersprachen oder genutzte Middleware sind auf dieser Betrachtungsebene zu finden. Aspekte wie Schnittstellen oder Individual- vs. Standardsoftware haben ebenso Auswirkungen auf die Betrachtungsebenen "Was?" und "Warum?".

Wirtschaftliche Aspekte sind, ebenso wie planerische Aspekte, Querschnitts-Aspekte, die auf allen Betrachtungsebenen eine Rolle spielen. Strategische Entscheidungen ("Warum?") haben Kosten zur Folge, die sich in die Ebenen "Was?" und "Wie?" niederschlagen.

Operative Aspekte sind auf der Ebene "Wie?" anzuordnen. Betriebsort oder Abhängigkeiten bei Batch-Programmen werden hier adressiert. Die Ebenen "Warum?" und "Was?" kommen durch organisatorische Gesichtspunkte ebenso in Betracht, da eine Geschäftsstrategie, die beispielsweise eine Veränderung des Auslandsgeschäftes zur Folge hat, auch bei operativen Aspekten eine Rolle spielt.

Für die einzelnen Gruppen von Aspekten und Betrachtungsebenen geeignete Darstellungsformen zu entwickeln und diese zusammen mit den Projektpartnern hinsichtlich der Aufgabenstellung zu evaluieren, ist Teil unseres Forschungsprojektes.

5 Existierende Modelle und Werkzeuge

Existierende Modelle der Informatik und insbesondere aus dem Bereich des Software-Engineerings sollen mit der Softwarekartographie verträglich sein. Bereits vorhandene und etablierte Modelle und Visualisierungssprachen sollen nicht ignoriert, sondern sinnvoll eingesetzt und integriert werden.

Die für die Softwarekartographie relevanten Modelle werden in Abschnitt 5.1 auf ihren Nutzen hin untersucht und gegenüber der Softwarekartographie abgegrenzt. Abschnitt 5.2 gibt einen Überblick über existierende Werkzeuge, die einen Bezug zur Beschreibung von Anwendungslandschaften haben.

5.1 Modelle

In der Informatik und den Wirtschaftswissenschaften sind verschiedene Beschreibungssprachen und Notationen entstanden, die in den Bereichen des Software-Engineerings eingesetzt werden, insbesondere die *Unified Modeling Language* und die *Ereignisgesteuerten Prozessketten* sind hervorzuheben und werden in den folgenden Abschnitten in einen Zusammenhang zur Softwarekartographie gebracht.

5.1.1 UML - Unified Modeling Language

Die *Unified Modeling Language*[7] (UML) ist der *de facto Standard* zur Beschreibung und Modellierung von Softwaresystemen. Das Hauptaugenmerk der UML mit seinen verschiedenen Diagrammtypen liegt auf einem *einzelnen* Softwaresystem:

> The Unified Modeling Language (UML) is a graphical language for visualizing, specifying, constructing, and documenting the artifacts of a software-intensive system. The UML offers a standard way to write a system's blueprints, including conceptual things such as business processes and system functions as well as concrete things such as programming language statements, database schemas, and reusable software components. [OMG03b]

Dem obigen Zitat ist anhand der Wortfolge "*a* software-intensive system" zu entnehmen, dass UML sich auf einzelne Softwaresystemen konzentriert und nicht auf die Beschreibung und Modellierung von Anwendungslandschaften, die aus vielen Softwaresystemen bestehen. Die optionale Systemgrenze im Anwendungsfalldiagramm manifestiert diese These.

Soll UML zur Beschreibung von Anwendungslandschaften verwendet werden, so könnten einzelne Diagrammtypen folgendermaßen verwendet werden:

[7] Derzeit ist Spezifikation 1.5 die aktuell abgeschlossene Version, UML 2.0 befindet sich kurz vor der Finalisierung.

Anwendungsfalldiagramm: Das Anwendungsfalldiagramm stellt Beziehungen zwischen Akteuren und Anwendungsfällen dar, Informationssysteme würden durch die Systemgrenze beschrieben und die Anwendungsfälle den einzelnen Systemen zugeordnet. Akteure wären Nutzer (z.B. Personen oder Personengruppen) von Informationssystemen oder Organisationseinheiten denen Informationssysteme zugeordnet würden. Technische, wirtschaftliche oder planerische Aspekte, wie beispielsweise Programmiersprachen oder Betriebskosten, sind durch Anwendungsfälle nicht abbildbar.

Anwendungslandschaften mit hundert oder mehr Informationssystemen mit Anwendungsfalldiagrammen zu modellieren ist nicht zielführend, da die in Abschnitt 4 beschriebenen Aspekte nicht intuitiv abgebildet werden könnten.

Klassen-/Objektdiagramm: Die grundlegenden Elemente Klasse, Assoziation, Vererbung und Rolle des Klassendiagramms könnten zur Modellierung von Anwendungslandschaften genutzt werden, Stereotypen für Klassen und Assoziationen könnten bestimmte Ausprägungen definieren. Der Stereotyp "InformationSystem" bei einer Klasse würde bedeuten, dass diese Klasse ein Informationssystem repräsentiert; "OrganizationalUnit" würde einer Klasse die Bedeutung einer Organisationseinheit geben.

Schnittstellen-Beziehungen zwischen Systemen könnten durch Assoziationen in Verbindung mit dem Lollipop-Symbol dargestellt werden. Ein zusätzlicher Stereotyp an der Assoziation kann kennzeichnen, ob es sich beispielsweise um eine "Online-" oder "Batch-Schnittstelle" handelt.

Auf diesem Wege könnten die relevante Aspekte Informationssystem, Schnittstelle und Organisationseinheit in einem Objektdiagramm visualisiert werden. Die begrenzte Anzahl graphischer Elemente des Klassen-/Objektdiagramms erlaubt es jedoch nicht, verschiedene Elemente für Geschäftsprozesse, Informationssysteme oder Organisationseinheiten zu verwenden. Ebenso würde die Komplexität des Diagramms durch alle eingezeichneten Schnittstellen eine intuitive Darstellung verhindern, da beispielsweise 100 Informationssysteme mit 300 Schnittstellen-Beziehungen zu 300 Assoziationslinien führen würde.

Komponentendiagramm: Das Komponentendiagramm ist eines der beiden Implementierungsdiagramme von UML und visualisiert die Verteilung von Softwarekomponenten auf verschiedene Laufzeitobjekte (z.B. Applikationsserver oder Datenbankserver). Da physikalische Geräte und Middleware-Komponenten relevante Aspekte für Softwarekarten sind, können Komponentendiagramme genutzt werden, um auf einem detaillierten Level für einzelne Softwaresysteme Informationen über ihre physikalische Verteilung zu erhalten. Ein "Zoom In" auf einer Softwarekarte könnte somit bei einem höherem Detail-Level ein Komponentendiagramm als "Subdiagramm" für ein Informationssystem zeigen.

Die obigen Beispiele zur Verwendung von UML zur Beschreibung von Anwendungslandschaften lassen sich auf die anderen Diagrammtypen übertragen. Allen gemein ist, dass die Anzahl der graphischen Elemente zur Beschreibung der relevanten Aspekte nicht ausreichend ist, um intuitive Darstellungen zu erhalten, da für mehrere Aspekte nur wenige graphisch unterschiedliche Elemente existieren. Ebenso definiert das UML Meta-Modell keine Semantik zur Verortung von Elementen in einem Diagramm oder erlaubt es, der Größe eines Elementes eine bestimmte Semantik zu geben. Inwieweit eine Erweiterung von UML 2.0 [OMG03a] durch die *UML Profiles* möglich ist und die Anforderungen an Softwarekarten sinnvoll abgebildet werden können, muss in der kommenden Projektphase analysiert werden, wenn das konzeptuelle Modell für Softwarekarten einen Abgleich mit den *UML Profiles* erlaubt.

Ein existierender Ansatz zur Nutzung von UML mit Bezug zu Anwendungslandschaften, bei dem das UML Meta-Modell erweitert wird, ist ARCUS [HMT02]. UML wird hierbei um spezifische Stereotypen, Einschränkungen und Notationen erweitert, um verschiedene Aspekte darstellen zu können. ARCUS definiert die vier Ebenen Prozess-, Fachbegriffs-, Anwendungs- und Systemarchi-

tektur, die miteinander über Assoziationen verbunden sind und visualisieren verschiedene Ebenen einer Anwendungslandschaft.

Im Gegensatz zu ARCUS soll die Softwarekartographie nicht nur verschiedene Ebenen miteinander verbinden, sondern mehrere relevante Aspekte auf einer Softwarekarte darstellen. ARCUS bietet beispielsweise nicht die Möglichkeit, Geschäftsprozesse und Informationssysteme auf einer Ebene zu visualisieren. Wirtschaftliche oder planerische Aspekte finden in ARCUS bei der Darstellung und im Modell keine Berücksichtigung.

5.1.2 EPK - Ereignisgesteuerte Prozessketten

Die ereignisgesteuerten Prozessketten dienen der Geschäftsprozessmodellierung, -optimierung und -analyse. In der ursprünglichen Definition [KNS92] wurden ausschließlich "Ereignisse" und "Funktionen" verwendet, die über verschiedene Verknüpfungsoperatoren verbunden wurden, so dass relevante Aspekte für Softwarekarten nicht abgebildet werden könnten, da kein Element zur Modellierung des zentralen Elementes "Informationssystem" existiert.

Weiterentwicklungen der ereignisgesteuerten Prozessketten finden sich beispielsweise im SAP R/3 System und dem ARIS Toolset der IDS Scheer AG. Das ARIS Toolset mit den verschiedenen ARIS-Sichten (Funktions-, Organisations-, Daten-, Leistungs und Steuerungssicht) definiert verschiedene Modelle und Objekte, wobei die einzelnen Sichten miteinander in Beziehung gesetzt werden. Das ARIS Toolset bietet ca. 100 verschiedene Modelltypen und über 200 Objekttypen.

Neben den reinen Elementen zur Prozessmodellierung, wie Ereignisse, Funktionen, Organisationseinheiten etc., existieren im ARIS Toolset ebenso Elemente zur Modellierung von Anwendungslandschaften wie beispielsweise das Element "Anwendungssystem". Die Diagrammart "erweiterte Ereignisgesteuerte Prozesskette (eEPK)" bietet zwar dieses Element "Anwendungssystem", ist jedoch zur Modellierung von Anwendungslandschaften nur bedingt geeignet, da der Fokus und die Elemente klar auf Prozessketten liegen.

Einige unserer Projektpartner nutzen bereits das ARIS Toolset, um ihre Anwendungslandschaft zu erfassen und zu modellieren. Im Rahmen einer Forschungsarbeit lassen wir derzeit das ARIS Toolset auf seine Möglichkeiten zur Darstellung von Softwarekarten untersuchen.

5.1.3 Modell zur Softwarekartographie

Aus existierenden Ansätzen für Softwarekarten (siehe Abschnitt 3) und den Aspekten, die auf Softwarekarten visualisiert werden sollen (siehe Abschnitt 4), lassen sich erste Anforderungen an die Visualisierungssprache für die Softwarekartographie zusammenstellen:

- Die Position eines Elementes auf einer Softwarekarte kann eine bestimmte Semantik erhalten, wobei eine absolute Position, die sich durch ein hinterlegtes Koordinatensystem ergibt, und eine relative Position, die durch die Distanz zweier Objekte zueinander entsteht, unterschieden werden kann. Die Softwarekarten C und D (siehe Abschnitt 3) verwenden eine absolute Positionierung innerhalb eines 2-dimensionalen Koordinatensystems, das bei Softwarekarten C und D durch den Prozess auf der X-Achse und unterschiedliche Aspekte entlang der Y-Achse ergibt.

  Die relative Position ergibt sich durch die Distanz zweier Elemente zueinander. Vergleichbar ist dies mit *Self Organizing Maps* (SOMs) [Koh97], bei der nahe beieinander liegende Punkte eine stärkere Verwandtschaft bedeutet, als eine größere Distanz zwischen den Punkten[8]. Bei Darstellungen, die ausschließlich mit der relativen Position arbeiten, hat die Position eines Elementes allein keine Semantik, da kein Koordinatensystem hinterlegt ist.

[8] Die Distanz ergibt sich bei SOMs aus den hinterlegten Algorithmen, so dass je nach Wahl des Algorithmus die Distanz eine bestimmte Semantik erhält.

Ein Spezialfall der relativen Position ergibt sich, wenn eine Element von einem anderen *umschlossen* wird analog zu einem UML-Komponenten oder Verteilungsdiagramm, bei dem beispielsweise Komponenten zu Laufzeitobjekten zugeordnet werden, indem eine Komponente in einem umgebenden Laufzeitobjekt positioniert wird. Softwarekarten A, B und C nutzen dieses Umschließen ebenso um logische Einheiten zu bilden oder Datenbanken den Systemen zuzuordnen.

- Die Form eines Elementes kann für unterschiedliche Objekttypen verwendet werden. Die Softwarekarten aus Abschnitt 3 nutzen beispielsweise Chevren für Prozessschritte und Datentöpfe für Datenbanken, wobei andere Elemente wie das Rechteck beispielsweise mehrfach belegt sind (System, Applikation, Funktionsbereich etc.). Die Anzahl von Elementen mit unterschiedlicher Form ist jedoch bei den Softwarekarten A bis D relativ gering, so dass ein breiteres Spektrum an Formen für unterschiedliche Objekttypen wie Informationssysteme, Schnittstellen-Konnektoren, Nutzergruppen etc. verwendet werden kann.

- Die Farbe und der Tonwert eines Elementes auf einer Softwarekarte kann eine bestimmte Semantik erhalten, wobei die Farbe bzw. der Tonwert des Hintergrunds, die Schraffur und die Rahmenfarbe unterschiedlich genutzt werden kann. Die Softwarekarten aus Abschnitt 3 nutzen Farben zur Darstellung von unterschiedlichen Attributen, Softwarekarte C nutzt beispielsweise die Hintergrundfarbe für verschiedene Status der Informationssysteme, während Softwarekarte A den Farbcode für die Funktionsbereiche unternehmensweit einsetzt und eine hohen Wiedererkennungswert erreicht.

  Die existierenden Beispiele nutzen Farben und Tonwerte jedoch nicht, um Skalen zu hinterlegen und quantifizierbare Aspekte zu kodieren. Mittels der Verwendung von Farben und Tonwerten könnten beispielsweise Wartungskosten von Systemen auf die gleiche Kartengrundlage wie Softwarekarte A projiziert werden.

- Die Größe eine Elementes auf einer Softwarekarte kann eine bestimmte Semantik erhalten. Durch das Variieren der Größe von Elementen können quantifizierbare Aspekte dargestellt werden, Blasen-Diagramme (z.B. MS Excel, ARIS Toolset) oder Landkarten, die Einwohnerzahlen von Städten durch unterschiedlich große Kreise darstellen, sind existierende Beispiele.

Analog zu UML werden sich verschiedene Arten von Diagrammen – sprich Softwarekarten – herauskristallisieren, die unterschiedliche der obigen Anforderungen erfüllen müssen. Inwieweit ebenso verschiedene Kartengründe[9] entwickelt werden, auf die dann verschiedene Schichten, die einzelne Aspekte enthalten, aufgetragen werden, wird sich im weiteren Projektverlauf ergeben.

5.2 Werkzeuge

In den Bereichen Geschäftsprozess-, (Unternehmens-)Datenmodellierung und IT-Management existieren verschiedene Werkzeuge, die die entsprechende Anwendungsdomäne unterstützen sollen.

Bei unseren Recherchen sind wir insbesondere auf die Werkzeuge in Tabelle 1 aufmerksam geworden. Zum jetzigen Zeitpunkt befassen wir uns intensiver mit dem SITM Solution Framework und dem ARIS Toolset; eine abschließende Analyse der Werkzeuge aus Tabelle 1 für die Softwarekartographie steht noch aus.

[9] In der Kartographie wird bei einer thematischen Karte der Kartengrund durch eine topograpische Karte gebildet. In der Softwarekartographie ist das Analogon der Grundaufbau einer Softwarekarte, bei der beispielsweise Funktionsbereiche oder Geschäftsprozesse mit Nutzergruppen die Basis sind.

Name des Herstellers	Name des Werkzeuges	URL
Adaptive	Adaptive	http://www.adaptive.com
alfabet meta-modeling	SITM Solution Framework	http://www.alfabet.de
ASG Software Solutions	Rochade	http://www.viasoft.com/
Casewise	Casewise	http://www.casewise.com
Computas	Metis	http://www.computas.com
IDS-Scheer	ARIS Toolset	http://www.ids-scheer.de
Méga International	Méga	http://www.mega.com
Popkin	Software System Architect	http://www.popkin.com
Proforma	Provision Modeling Suite	http://www.proformacorp.com
Ptech	Ptech Enterprise Framework	http://www.ptechinc.com
Softlab	Softlab Enabler	http://www.softlab.com/
Troux	Troux	http://www.troux.com
Visible Systems	Visible Advantage	http://www.visible.com

Tabelle 1: Werkzeuge aus den Domänen Geschäftsprozess-/Datenmodellierung und IT-Manangement

6 Zusammenfassung und Ausblick

Die Softwarekartographie als Disziplin zur Beschreibung, Bewertung und Gestaltung komplexer Informations-Infrastrukturen soll die verschiedenen Betrachtungsebenen für Anwendungslandschaften (siehe Abschnitt 2) auf Softwarekarten transferieren. Die unterschiedlichen Visualisierungen sollen planerische, fachliche, technische, wirtschaftliche und operative Aspekte enthalten, die alleinstehend und durch geeignete Kombinationen Informationen über die Anwendungslandschaft repräsentieren. Das zentrale Objekt der Betrachtung sind die betrieblichen Informationssysteme der Anwendungslandschaft, die als Ganzes die Wertschöpfungskette eines Unternehmens unterstützt.

Die existierenden Softwarekarten einzelner Projektpartner (siehe Abschnitt 3) zeigen, dass ein Bedarf zur Visualisierung von Anwendungslandschaften besteht und dass Unternehmen unterschiedliche Aspekte auf diesen Softwarekarten darstellen wollen. Die zusätzlich aufgenommen Anforderungen an Softwarekarten (siehe Abschnitt 4 und Anhang A) müssen in den nächsten Schritten in einen ganzheitliches, konzeptuelles Modell umgesetzt werden.

Parallel zu dem Entwurf eines Modells für die Softwarekartographie sind geeignete Darstellungsformen für die einzelnen Aspekte zu finden, um diese mit den Projektpartnern zu diskutieren und deren Praxisrelevanz zu überprüfen.

Die existierenden Softwarekarten in den einzelnen Unternehmen sind nur teilweise mittels einer Werkzeugunterstützung erzeugt worden, wobei die Erhebung der relevanten Daten größtenteils per Hand vorgenommen wurde. Sollen Softwarekarten als Steuerungsinstrument für Anwendungslandschaften genutzt werden und einen dauerhaften Wertbeitrag liefern, so ist ein geeignetes Repository, ein Prozess zur steten Aktualisierung der Daten und eine (Semi-)Automatisierung des Visualisierungsprozesses unumgänglich.

Die Analyse der Werkzeuge im Umfeld der Softwarekartographie (siehe Abschnitt 5.2) muss zeigen, inwieweit existierende Anwendungen als geeignetes Repository dienen können und den Prozess der Datenerhebung und -pflege geeignet unterstützen. Die Entwicklung eines geeigneten Vorgehensmodells, sowohl für die initiale Erhebung als auch für die kontinuierliche Pflege und Erweiterung der Daten, ist eine weitere Herausforderung in dem Forschungsprojekt Softwarekartographie.

Das dynamische Verhalten der Anwendungslandschaft, welches sich sowohl durch den täglichen Betrieb von Informationssystemen als auch durch die stete Weiter- und Neuentwicklung selbiger ergibt, durch geeignete Softwarekarten zu unterstützen und ein Vorgehensmodell zur Erstellung und Pflege dieser Softwarekarten zu entwickeln ist unser Ziel.

A Relevante Aspekte für Softwarekarten

Im Folgenden werden die verschiedenen relevanten Aspekte gelistet und den einzelnen Kategorien (siehe Abschnitt 4) zugeordnet. Neben dem Namen des Aspektes, der Kategorie und einer Beschreibung wird die Art des Aspektes *quantifizierbar* und/oder *nicht quantifizierbar* angegeben. Quantifizierbare Aspekte können mittels Metriken und Kennzahlensystemen für numerische Auswertungen herangezogen werden.

Der Aufbau von Kennzahlensystemen für die quantifizierbaren Aspekte soll in einem zweiten Schritt erfolgen. Existierende Kennzahlen und Kennzahlensysteme (vgl. [Küt03]) müssen auf ihre Anwendbarkeit für Softwarekarten hin untersucht und die Ergebnisse validiert werden.

A.1 Planerische Aspekte

Name	Programme und Projekte auf der Anwendungslandschaft
Kategorie	Planerisch
Art	Nicht quantifizierbar
Beschreibung	Zur Programm-/Projektplanung, zum Aufzeigen von Abhängigkeiten zwischen Programmen/Projekten sollen Softwarekarten aktuelle und geplante Programme/Projekte in Verbindung mit den betroffenen Informationssystemen visualisieren. Durch die starke Verflechtung einer konzernweiten Anwendungslandschaft können Abhängigkeiten durch eine überblicksartige Darstellung besser erkannt und die betroffenen Programme/Projekte benannt werden. Die Softwarekarten sollen mehrere Planungsphasen berücksichtigen.

Name	Lebenszyklus eines Informationssystems
Kategorie	Planerisch
Art	Nicht quantifizierbar
Beschreibung	Der Lebenszyklus eines Informationssystems beschreibt die verschiedenen Phasen eines Informationssystems (in Entwicklung, im Test, in Produktion, in Ablösung, abgelöst) und soll auf Softwarekarten visualisiert werden. Durch das Knüpfen einer Version an das Informationssystem können beliebig viele Lebenszyklen für ein Informationssystem existieren.

Name	Zeitliche Veränderung der Anwendungslandschaft
Kategorie	Planerisch
Art	Nicht quantifizierbar
Beschreibung	Durch die Langlebigkeit der Informationssysteme ist die zeitliche Veränderung der Anwendungslandschaft von besonderem Interesse. Die geplanten und getätigten Veränderungen der Anwendungslandschaft durch verschiedene IT-Projekte sollen auf Softwarekarten dargestellt werden. Durch das Überlagern zweier Softwarekarten mit gleichen Aspekten unterschiedlichen Datums kann das Erreichen von strategischen Zielen oder Projektzielen belegt werden.

Name	Zielarchitektur der Anwendungslandschaft
Kategorie	Planerisch
Art	Nicht quantifizierbar
Beschreibung	Um strategische Ziele, beispielsweise das Reduzieren der Anzahl von Individualsoftware, auf Softwarekarten zu visualisieren, sollen strategische Ziele (ggf. ohne Bezug zu einem Projekt oder Programm) auf Softwarekarten visualisiert werden können.

A.2 Fachliche Aspekte

Name	Gruppierung von Informationssystemen zu logischen Einheiten
Kategorie	Fachlich
Art	Nicht quantifizierbar
Beschreibung	Die Gruppierung von Informationssystemen zu logischen Einheiten (z.B. Funktionseinheiten, Organisationseinheiten) soll auf Softwarekarten möglich sein.

Name	Nutzungsintensität je Informationssystem
Kategorie	Fachlich
Art	Quantifizierbar
Beschreibung	Die Nutzungsintensität von Informationssystemen beispielsweise die Nutzungszeit pro Benutzer pro Tag oder die Anzahl von Geschäftsvorfällen pro Nutzer (siehe auch *Transaktionsraten und Datenvolumen je Informationssystem*) sollen auf Softwarekarten visualisiert werden. Die Softwarekarten können für Statistiken oder zur Erkennung von hochfrequentierten Informationssystemen genutzt werden. Eine Kombination mit wirtschaftlichen Aspekten kann zur einer Kosten/Nutzen-Funktion führen.

Name	Anzahl von Nutzern je Informationssystem
Kategorie	Fachlich
Art	Quantifizierbar
Beschreibung	Die Anzahl von Nutzern (registrierte Nutzer etc.) ist neben anderen Faktoren, wie beispielsweise die Transaktionsrate, ein Mittel zur Bewertung der Bedeutung eines Informationssystems. Desweiteren kann die Anzahl von Nutzern (ebenso wie die *Nutzungsintensität je Informationssystem*) für eine Kosten/Nutzen-Funktion verwendet werden.

Name	Nutzergruppen/Arbeitsplätze je Informationssystem
Kategorie	Fachlich
Art	Quantifizierbar/nicht quantifizierbar
Beschreibung	Die Nutzergruppen können Aufschluss über die Arten von Arbeitsplätzen in einem Unternehmen geben. Diese Zuordnung von Nutzergruppen zu Informationssystemen hilft bei der Bildung von logischen Einheiten und kann sinnvolle Gruppierungen durch verwandte Nutzungsszenarien aufzeigen. Zusätzlich soll die Anzahl von Nutzergruppen je Informationssystem als Messwert visualisiert werden.

Name	Funktion je Informationssystem
Kategorie	Fachlich
Art	Quantifizierbar
Beschreibung	Mittels eines Messwertes wie beispielsweise *Function Points* soll die Funktionalität der Informationssysteme quantifiziert und auf Softwarekarten abgebildet werden. Die zeitliche Veränderung dieses Messwertes (steigend/fallend/konstant) in Kombination mit der Veränderung der Wartungskosten kann als Indikator für ein mögliches Reengineering oder eine Performanzoptimierung genutzt werden.

Name	Unterstützte Geschäftsprozesse je Informationssystem
Kategorie	Fachlich
Art	Nicht quantifizierbar
Beschreibung	Die Zuordnung von Funktionen – im Sinne der unterstützenden Geschäftsprozesse – zu Informationssystemen soll auf Softwarekarten visualisiert werden. In Kombination mit logischen Einheiten sollen Informationssysteme mit gleichem funktionalen Charakter in unterschiedlichen Einheiten erkannt werden.

Name	Geschäftsobjekte je Informationssystem
Kategorie	Fachlich
Art	Nicht quantifizierbar
Beschreibung	Durch die große Anzahl von Informationssystemen einer Anwendungslandschaft existieren in unterschiedlichen Informationssystemen dieselben Geschäftsobjekte. Softwarekarten sollen diese redundante Datenhaltung visualisieren und ebenso die für einzelne Geschäftsobjekte *führenden* Informationssysteme hervorheben. Zusätzlich sollen Geschäftsobjekte, die über Schnittstellen ausgetauscht werden in Kombination mit dem Aspekt *Schnittstellen-Beziehungen* dargestellt werden.

A.3 Technische Aspekte

Name	Transaktionsraten je Informationssystem
Kategorie	Technisch
Art	Quantifizierbar
Beschreibung	Die Transaktionsrate im Sinne von Geschäftstransaktionen (z.B. Überweisungen pro Minute) soll die Nutzung eines Informationssystems im technischen Sinne (siehe *Nutzungsintensität*) abbilden. Wird zusätzlich eine zeitliche Komponenten (24-Stundenskala, Monatsskala etc.) verwendet, so können beispielsweise Lastzeiten erkannt werden. Wird die Anzahl von Transaktionen nicht auf Anwendungssystemebene gemessen, sondern auf Ebene eines Hardwaresystems und zusätzlich die Anzahl der Transaktionen pro Informationssystem erfasst, so kann dieses Maß zur Verrechnung der Kosten eines Hardwaresystems genutzt werden; insbesondere bei Host-Systemen, die eine große Anzahl von Anwendungen gleichzeitig betreiben, ist eine derartige Kostenrechnung relevant.

Name	Datenvolumen je Informationssystem
Kategorie	Technisch
Art	Quantifizierbar
Beschreibung	Das genutzte Datenvolumen eines Informationssystems zur Speicherung von Anwendungsdaten soll auf Softwarekarten visualisiert werden. Der Nutzen dieser Softwarekarten und dieses Aspektes ist vergleichbar mit *Transaktionsraten je Informationssystem*. Statistische Auswertungen mit zeitlichem Verlauf der Datenvolumen und die Zuordnung von Datenvolumen je Informationssystem in einem Datenbanksystem sollen dargestellt werden.

Name	Individual- vs. Standardsoftware
Kategorie	Technisch
Art	Nicht quantifizierbar
Beschreibung	Unternehmen, die es beispielsweise als strategisches Ziel verfolgen, die Zahl von Individualsoftwaresystemen zu reduzieren, können diese Softwarekarten zur Ziel-Verfolgung nutzen. Insbesondere die Betrachtung von Informationssystemen in Funktionsbereichen, die geprägt sind durch Gesetze und Regeln (z.B. Finanzbuchhaltung, Controlling, Personalwesen etc.), kann Missstände aufzeigen. Zusätzlich kann bei Individualsoftware zwischen Eigen- und Fremdentwicklung bzw. einer Kombination unterschieden werden, um eine feinere Detaillierung zu erreichen und zusätzlich Abhängigkeiten von externem Know-how bei Veränderungen an einem oder umgebenden Informationssystemen zu erkennen.

Name	Schnittstellen-Beziehungen
Kategorie	Technische
Art	Quantifizierbar/Nicht quantifizierbar
Beschreibung	Durch die Starke Vernetzung einer Anwendungslandschaft sind für Softwarekarten die Beziehungen von Informationssystemen relevant, sowohl die einzelnen Kommunikationswege als auch die Art der Kommunikation müssen betrachtet werden. Bei den Kommunikationswegen sind die Kommunikationspartner (z.B. System A kommuniziert mit System B) sowie die Kommunikationsrichtung (rufendes/gerufenes System) zu erfassen. Die Kommunikationsarten (Online vs. Batch vs. Manuell), die Art des Zugriffs (lesend vs. schreibend) und Implementierungstechnologien der Schnittstellen-Konnektoren müssen unterschieden werden (siehe *Schnittstellen-Konnektoren*).

Name	Schnittstellen-Konnektoren
Kategorie	Technisch
Art	Nicht quantifizierbar
Beschreibung	Um die Schnittstellen-Beziehungen detaillierter zu betrachten, sind die Kommunikationsarten zu betrachten, die sich aus der Implementierung der fachlichen Schnittstellen-Konnektoren ergeben. Die technischen Schnittstellen-Konnektoren implementieren fachliche Schnittstellen-Konnektoren mittels einer bestimmten Technologie, so dass hieraus die Kommunikationsart (z.B. *Remote Method Invocation*, Batch-Datei, Fax etc.) resultiert. Verwendet der technische Schnittstellen-Konnektor eine Middleware (z.B. IBM MQSeries), so ist dies geeignet zu berücksichtigen.

Name	Implementierungssprache je Informationssystem
Kategorie	Technisch
Art	Quantifizierbar/nicht quantifizierbar
Beschreibung	Die Informationssysteme einer Anwendungslandschaft sind in verschiedenen Programmiersprachen (C, C++, Java, Cobol, Pascal, ABAP etc.) realisiert, wobei ein einzelnes Informationssystem in mehreren Programmiersprachen implementiert sein kann. Die Anforderung, Informationssysteme hinsichtlich der Implementierungssprachen zu betrachten, resultiert durch die Mannigfaltigkeit von Programmiersprachen in existierenden Anwendungslandschaften. In Kombination mit der Anforderung *Zeitliche Veränderung der Anwendungslandschaft* und *Individual- vs. Standardsoftware* ist erkennbar, wie sich die Anwendungslandschaft und das benötigte Know-how zur Wartung/Weiterentwicklung verändern. Der quantifizierbare Anteil dieses Aspektes ergibt sich, wenn die Häufigkeit der Implementierungssprachen betrachtet wird. Die Häufigkeit einer Implementierungssprache in einer Anwendungslandschaft lässt einen Rückschluss auf das benötigte Know-how der Anwendungsentwickler bzw. Externen zu, eine Erweiterung um *Lines of Code* je Implementierungssprache in einem Informationssystem erhöht die Aussagekraft.

Name	Softwarearchitektur je Informationssystem
Kategorie	Technisch
Art	Nicht quantifizierbar
Beschreibung	In Abhängigkeit von der Softwarearchitektur sollen die Informationssysteme auf Softwarekarten unterschieden werden, mögliche Ausprägungen sind: • 3-schichtige Client/Server-Architektur mit *Thin-Client*, Applikations- und Datenbankserver • 2-schichtige Client/Server-Architektur mit *Fat-Client* und Datenbankserver • Monolith Weitere Ausprägungen müssen entsprechend der spezifischen Anforderungen parametrisiert werden können.

Name	Genutzte Middleware je Informationssystem
Kategorie	Technisch
Art	Quantifizierbar/nicht quantifizierbar
Beschreibung	Die Verbindung von Informationssystemen und den genutzten Middleware-Systemen ist für Softwarekarten relevant, mögliche Ausprägungen sind beispielsweise *Web Application Server*, *Transaction Server* oder *Messaging Server*. Auf einer detaillierteren Ebene sollen bestimmte Technologien oder Hersteller (z.B. IBM, BEA, Oracle oder Siemens) zugeordnet werden. Des Weiteren sollen Abhängigkeiten von bestimmten Produkten und Versionen visualisiert werden können, um bei Migrationen die betroffenen Systeme zu identifizieren. Der quantifizierbare Anteil dieses Aspektes ergibt sich, wenn die Häufigkeit der auftretenden Middleware-Systeme betrachtet wird.

Name	Genutzte Datenbankmanagementsysteme je Informationssystem
Kategorie	Technisch
Art	Quantifizierbar/nicht quantifizierbar
Beschreibung	Informationssysteme nutzen verschiedene Datenbanktechnologien und Datenbankmanagementsysteme zur persistenten Speicherung von Daten. Zum Zweck der Homogenisierung der Anwendungslandschaft (Reduktion von Lizenzkosten, Know-how etc.) sollen unterschiedliche und gleiche Datenbankmanagementsysteme der Informationssysteme erkannt werden. Analog zu *Genutzte Middleware je Informationssystem* sollen ebenso Abhängigkeiten zu bestimmten Produkten und Versionen aufgezeigt werden können. Der quantifizierbare Anteil dieses Aspektes ergibt sich, wenn die Häufigkeit der auftretenden Datenbankmanagementsysteme betrachtet wird.

Name	Genutztes Betriebssystem je Informationssystem
Kategorie	Technisch
Art	Quantifizierbar/nicht quantifizierbar
Beschreibung	Zum Zweck der Homogenisierung der Anwendungslandschaft hinsichtlich der Betriebssysteme(Reduktion von Lizenzkosten, Know-how etc.) sollen unterschiedliche und gleiche Betriebssysteme der Informationssysteme erkannt werden. Analog zu *Genutzte Middleware je Informationssystem* sollen ebenso Abhängigkeiten zu bestimmten Betriebssystemen und Versionen aufgezeigt werden können. Der quantifizierbare Anteil dieses Aspektes ergibt sich, wenn die Häufigkeit der auftretenden Betriebssysteme betrachtet wird.

Name	Hardwaresystem je Informationssystem
Kategorie	Technisch
Art	Quantifizierbar/nicht quantifizierbar
Beschreibung	Zum Zweck der Homogenisierung der Anwendungslandschaft (Gemeinsame Ressourcennutzung, Know-how etc.) sollen unterschiedliche und gleiche Hardwaresysteme der Informationssysteme erkannt werden. Analog zu *Genutzte Middleware je Informationssystem* sollen ebenso Abhängigkeiten zu bestimmten Produkten aufgezeigt werden können. Der quantifizierbare Anteil dieses Aspektes ergibt sich, wenn die Häufigkeit der auftretenden Hardwaresysteme betrachtet wird.

Name	Skalierbarkeit und Fehlertoleranz je Informationssystem
Kategorie	Technisch
Art	Nicht quantifizierbar
Beschreibung	Softwarekarten sollen die Eigenschaften der Informationssysteme hinsichtlich Skalierbarkeit und Fehlertoleranz darstellen können. Ergänzend soll die tatsächliche Verteilung der einzelnen Informationssysteme berücksichtigt werden.

A.4 Wirtschaftliche Aspekte

Name	Kosten je Informationssystem
Kategorie	Wirtschaftlich
Art	Quantifizierbar
Beschreibung	Verschiedene Kostenarten (Wartungskosten, Bereitschaftskosten etc.) sollen auf Softwarekarten visualisiert werden. Neben den Kosten in einer bestimmten Periode sollen zeitliche Verläufe der einzelnen Kostenarten abgebildet werden können.

Name	Kapitalwert der Informationssysteme
Kategorie	Wirtschaftlich
Art	Quantifizierbar
Beschreibung	Eine Kapitalwertanalyse der Informationssysteme mit zeitlichem Verlauf des Kapitalwertes kann beispielsweise einen Handlungsbedarf für ein Reengineering aufzeigen. Softwarekarten sollen verschiedene zeitliche Verläufe und Ist-Werte visualisieren.

A.5 Operative Aspekte

Name	Ausfall- und Betriebszeiten je Informationssystem
Kategorie	Operativ
Art	Quantifizierbar
Beschreibung	Die nominale Anzahl von Systemausfällen, die Ausfallzeiten sowie die prozentuale Systemverfügbarkeit sollen auf Softwarekarten dargestellt werden. Eine Detaillierung wie bei einem *IT-Cockpit*, welches sämtliche Ausfallzeiten, Fehleranalysen und Performanzmessungen analysiert und archiviert, ist für Softwarekarten nur bedingt relevant, da derartige Anforderungen bereits durch geeignete Werkzeuge unterstützt werden.

Name	Geographische Position des Betriebsortes je Informationssystem
Kategorie	Operativ
Art	Quantifizierbar/nicht quantifizierbar
Beschreibung	Softwarekarten sollen die unterschiedlichen Betriebsorte der Systeme darstellen. Unternehmen, dessen Informationssysteme und Hardwaresysteme geographisch verteilt sind, wollen eine überblicksartige Darstellung der Anwendungslandschaft in Kombination mit der geographischen Position, wobei ein Detaillierungsgrad auf Länder- bzw. Stadt-Ebene als ausreichend angesehen wird. Der quantifizierbare Anteil ergibt sich, wenn die Anzahl von Informationssystemen pro Standort bzw. geographischer Position ermittelt wird.

Name	Geographische Darstellung der Nutzer je Informationssystem
Kategorie	Operativ
Art	Quantifizierbar/nicht quantifizierbar
Beschreibung	Analog zu den Betriebsorten sollen auch die geographischen Positionen der Nutzer visualisiert werden. In Kombination mit dem Aspekt *Geographische Position des Betriebsortes je Informationssystem* können Differenzen zwischen Nutzungs- und Betriebsort erkannt werden. Eine Betrachtung der zeitlichen Entwicklung dieser Beziehung ergänzt die Analyse. Der quantifizierbare Anteil ergibt sich analog zum Aspekt *Geographische Position des Betriebsortes je Informationssystem.*

Name	Abhängigkeiten zwischen Informationssystemen
Kategorie	Operativ
Art	Nicht quantifizierbar
Beschreibung	Durch die Abhängigkeit von Informationssystemen, die über Schnittstellen miteinander interagieren, soll eine Softwarekarte derartige Zusammenhänge visualisieren. Die Softwarekarten sollen Aufschluss darüber geben, welche Domino-Effekte ein Ausfall eines Informationssystems herbeiführt. Durch die unterschiedliche Kopplung der Informationssysteme (Online vs. Batch vs. Manuell) können Domino-Effekte zeitversetzt auftreten, so dass die Art der Kopplung einen Aufschluss über zeitgleiche bzw. -nahe oder zeitversetzte Ausfälle gibt.

Glossar

A

Anwendungslandschaft (AL) Die Anwendungslandschaft ist die Gesamtheit aller betrieblichen Informationssysteme in einem Unternehmen.

Aspekt Ein Aspekt beschreibt eine oder mehrere Eigenschaften von Informationssystemen.

H

Hardwarelandschaft Die Hardwarelandschaft ist die Gesamtheit aller physikalischen Komponenten (Server, Netzwerkkomponenten etc.) zur Unterstützung der Informationssysteme.

K

Kennzahl Eine Kennzahl erfasst Sachverhalte von Informationssystemen quantitativ und in konzentrierter Form. Beispiele für Kennzahlen sind "Anzahl von Nutzern" oder "Transaktionsrate pro Minute". Kennzahlen sind eine echte Teilmenge von Aspekten.

S

Schnittstelle Systeme kommunizieren über Schnittstellen miteinander. Schnittstellen spezifizieren die exportierten Funktionen eines Teils eines Systems (z.B. einer Komponente) und sind definiert durch die Semantik (Wirkung der Funktionen), Syntax (Formal- und Ergebnisparameter) und die Kommunikationsart (Online vs. Offline) der Schnittstelle. Es kann zwischen internen und externen Schnittstellen unterschieden werden, wobei interne Schnittstellen nur von dem System selbst genutzt werden können, externe hingegen von dem System selbst und anderen Systemen. Zwei Systeme kommunizieren somit über externe Schnittstellen miteinander.

Schnittstellen-Beziehung Eine Schnittstellen-Beziehung existiert zwischen dem Import-Konnektor des Systems A (Client) und dem Export-Konnektor eines Systems B (Server). Eine existierende Schnittstellen-Beziehung bedeutet, dass die Systeme A und B über eine Schnittstelle miteinander kommunizieren.

Schnittstellen-Konnektor Für Schnittstellen-Konnektoren existieren die folgenden beiden Varianten: Export- und Import-Konnektor. Ein fachlicher Export-Konnektor ist die Implementierung einer Schnittstelle. Ein technischer Export-Konnektor ist die Implementierung eines fachlichen Export-Konnektors mittels einer bestimmten Technologie. Für eine fachlichen Export-Konnektor können mehrere technische Export-Konnektoren existieren, die die gleichen Funktionen der Schnittstelle in unterschiedlichen Technologien implementieren. Ein fachlicher bzw. technischer Import-Konnektor importiert einen fachlichen bzw. technischen Export-Konnektor und ist auf der Seite des Clients die Implementierung zur Kommunikation mit dem Server.

Sicht Mit einer Sicht wird eine spezielle Ausprägung einer Softwarekarte, auf der bestimmte Kennzahlen dargestellt werden, bezeichnet. Verschiedene Sichten können analog zum Ebenen-Prinzip [HGM02] übereinandergelegt werden und zu anderen Softwarekarten kombiniert werden.

Softwarekarte Eine Softwarekarte ist eine graphische Repräsentation der gesamten Anwendungslandschaft oder Ausschnitte selbiger. Eine Softwarekarte setzt sich zusammen aus einer oder mehrerer Sichten, die verschiedene Aspekte visualisieren.

Softwarekartographie Die Softwarekartographie beschreibt die Modelle und Methoden zur Beschreibung und graphischen Darstellung von Anwendungslandschaften durch Softwarekarten.

Softwarelandschaft Die Softwarelandschaft ist die Gesamtheit aller Softwaresysteme eines Unternehmen. Im Gegensatz zu Anwendungslandschaften gehören zu einer Softwarelandschaft auch Applikationsserver, Datenbankmanagementserver, Dateiserver etc., die keine betrieblichen Informationssysteme sind (vgl. Anwendungslandschaft).

Literatur

[Bal00] Balzert, Helmut: *Lehrbuch der Software-Technik*. 2. Spektrum, 2000. – ISBN 3–8274–0480–0

[Dat00] Date, C. J.: *An Introduction to Database Systems*. 7. Addison-Wesley, 2000. – ISBN 0–201–38590–2

[FS99] Fowler, Martin ; Scott, Kendall: *UML Distilled*. 2. Addison-Wesley, 1999. – ISBN 0–201–65783–X

[HGM02] Hake, Günter ; Grünreich, Dietmar ; Meng, Liqui: *Kartographie*. 8. de Gruyter, 2002. – ISBN 3–11–016404–3

[HMT02] Heberling, Marcus ; Maier, Christoph ; Tensi, Thomas: Visual Modelling and Managing the Software Architecture Landscape in a large Enterprise by an Extension of the UML. In: *OOPSLA, Workshop DSVL*. Seattle, 2002

[KNS92] Keller, G. ; Nüttgens, M. ; Scheer, A.-W.: Semantische Prozeßmodellierung / Institut für Wirtschaftsinformatik, Universität des Saarlandes. 1992. – Forschungsbericht. EPK

[Koh97] Kohonen, Teuvo: *Self-organizing maps*. 2. Berlin : Springer, 1997. – ISBN 3–540–62017–6

[Küt03] Kütz, Martin: *Kennzahlen in der IT : Werkzeuge für Controlling und Management*. 1. Heidelberg : dpunkt.Verlag, 2003. – ISBN 3–89864–225–9

[OMG03a] OMG. *UML 2.0 Superstructure Specification - Adopted Specification ptc/03-08-02*. 2003

[OMG03b] OMG. *Unified Modelling Language Specification, Version 1.5*. 2003

[Sch95] Scheer, August-Wilhelm: *Wirtschaftsinformatik*. 6. Springer-Verlag, 1995. – ISBN 3–540–58872–8