Architekturbeschreibung von Anwendungslandschaften: Softwarekartographie und IEEE Std 1471-2000

Josef Lankes, Florian Matthes, André Wittenburg

Software Engineering betrieblicher Informationssysteme – I19
Ernst Denert-Stiftungslehrstuhl
Institut für Informatik
Technische Universität München
Boltzmannstraße 3
85748 Garching
{lankes, matthes, wittenbu}@in.tum.de

Abstract: Jenseits der Beschreibung einzelner Softwaresysteme gewinnt die Beschreibung von Anwendungslandschaften, bestehend aus hunderten oder tausenden von Informationssystemen, zunehmend an Bedeutung. In unserem Forschungsprojekt *Softwarekartographie* haben wir den *State of the Art* zur Beschreibung von Anwendungslandschaften in der Praxis aufgenommen und entwickeln in Zusammenarbeit mit unseren Projektpartnern Methoden und (graphische) Modelle zur Beschreibung, Bewertung und Gestaltung von Anwendungslandschaften [MW04a; MW04b; LMW05].

In diesem Beitrag untersuchen wir den Begriffsapparat des IEEE Std 1471-2000 „IEEE Recommended Practice for Architectural Description of Software-Intensive Systems“ systematisch auf seine Eignung zur Architekturbeschreibung komplexer Anwendungslandschaften. Der Nutzen aus den standardisierten Begriffsdefinitionen des IEEE 1471 sowie den dort vorgestellten Best-Practice-Vorgehensweisen soll auch für das Management von Anwendungslandschaften zugänglich gemacht werden. Dazu identifizieren wir Stärken und Schwächen des Standards und schlagen eine Erweiterung des Begriffsapparats vor, um insbesondere den kritischen Prozess der Informationsbeschaffung adäquat zu modellieren.

1 Motivation

Die Komplexität von Anwendungslandschaften und der steigende Kostendruck in der Informationstechnologie haben Unternehmen in den letzten Jahren vor neue Herausforderungen gestellt. Eine dieser Herausforderungen liegt in dem Wandel der Anwendungslandschaft, die sich aus den betrieblichen Informationssystemen eines Unternehmens zusammensetzt. Das Management und die Planung der Anwendungslandschaft sollen existierende Investitionen sichern, neue Investitionen planen und nichts desto trotz Kosten reduzieren; eine Voraussetzung hierfür ist eine geeignete Dokumentation.

In unserem Forschungsprojekt *Softwarekartographie* entwickeln wir in Zusammenarbeit mit unseren Projektpartnern (u.a. BMW, Deutsche Börse, HVB Systems, T-Com)

Methoden und Modelle zur Beschreibung, Bewertung und Gestaltung von Anwendungslandschaften. Wir haben den *State of the Art* in der industriellen Praxis zur Dokumentation und Visualisierung von Anwendungslandschaften aufgenommen und die Anforderungen an geeignete Notationen und Modelle erfasst [MW04a; MW04b; LMW05]. Derzeit konzentrieren wir uns auf folgende Aspekte:

- Aufbau eines einheitlichen Begriffsapparates für Anwendungslandschaften
- Entwurf eines geeigneten Informationsmodells
- Visualisierung von Anwendungslandschaften in Form von Softwarekarten
- Entwicklung eines Prototypen zur Generierung von Softwarekarten

Bei der Entwicklung des Informationsmodells führen wir insbesondere die verschiedenen Anforderungen unserer Projektpartner derart zusammen, dass trotz des unterschiedlichen Informationsbedarfes ein gemeinsames Modell entsteht. Ausgangspunkt für die Entwicklung eines geeigneten Informationsmodells sind die *Stakeholder* und ihre *Interessen*, die in einem Informationsbedarf resultieren. Stakeholder können Organisationseinheiten (Fachbereiche etc.), reale Personen („Heinz Mustermann“) oder Rollen (CIO, Projektleiter, Systemverantwortlicher etc.) sein.

Folgendes Szenario illustriert diese Problemstellung: Ein Projektleiter will die Integrationsplanung seines Projektes in die existierende Anwendungslandschaft durchführen und analysiert hierzu die existierenden Schnittstellen eines anzubindenden CRM-Systems. Der Ausgangspunkt für seine Analyse darf jedoch nicht ausschließlich der Ist-Zustand zum Analysezeitpunkt sein, sondern muss Veränderungen über die Projektlaufzeit im CRM-System und in anderen betroffenen Systemen berücksichtigen. Dies könnte zu folgender Fragestellung führen: „Welche Informationssysteme, die mit meinem Projekt in Bezug stehen, werden während meines Projektes verändert und wie?“

Folgend stellen wir in Abschnitt 2 unseren Ansatz zum Aufbau geeigneter Sichten (*Views*) für Stakeholder, deren Interessen und Fragestellungen in Form von Softwarekarten vor und diskutieren die Anforderungen an derartige Sichten und ihre Verwendung. Im Anschluss (Abschnitt 3) stellen wir Terminologie und Konzepte des IEEE Std 1471-2000 (im Folgenden IEEE 1471) vor und diskutieren Stärken und Schwächen des Standards. Abschnitt 4 stellt eine Verfeinerung des IEEE 1471 für das Management von Anwendungslandschaften und die Softwarekartographie dar.

2 Fragestellungen und Softwarekarten

Sollen für die Fragestellungen der Stakeholder, die sich aus deren Interessen ergeben, geeignete Sichten entwickelt werden, so müssen die zur Beantwortung der Fragestellungen notwendigen Informationsobjekte (z.B. Geschäftsprozesse, Informationssysteme, Nutzungsorte; vgl. auch relevante Aspekte [MW04a]) entsprechend aufbereitet werden:

Die Fragestellung „Welche Geschäftsprozesse werden von welchen Informationssystemen unterstützt und wo werden diese Informationssysteme genutzt?“ resultiert in den drei Informationsobjekten *BusinessProcess*, *InformationSystem* und *Location*. Werden die Objekte geeignet in Beziehung gesetzt, entsteht das Modell in Abbildung 1.

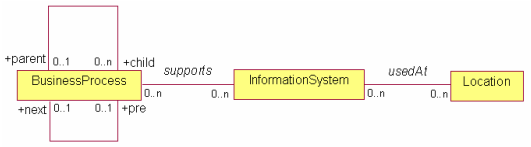

Abbildung 1: Exemplarisches Informationsmodell für eine Fragestellung

Um die ursprüngliche Fragestellung geeignet zu beantworten, sind prinzipiell zwei Ansätze denkbar: Zum einen können die Informationen in einer Berichtsform (*Report*) aufbereitet werden, wobei tabellarische Darstellungen und Diagramme (Balken-, Kreisdiagramme etc.) vorwiegend zum Informationstransport verwendet werden. Zum anderen können graphische Darstellungen erstellt werden, die unter Verwendung verschiedener Gestaltungsmittel [HGM02] (Linien, Flächen, Symbole etc.) unterschiedliche Informationsobjekte darstellen und somit die Informationen transportieren.

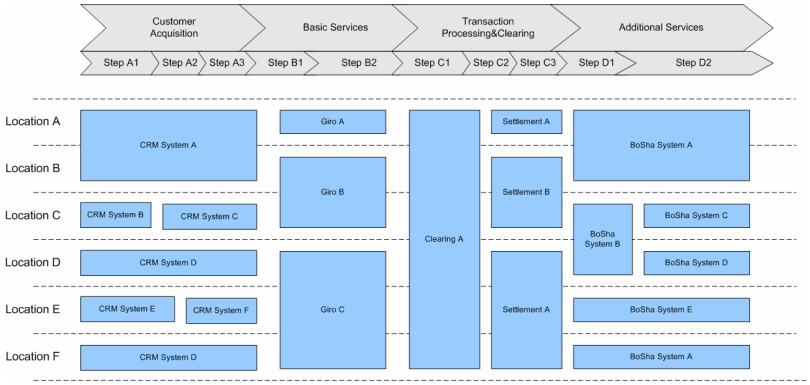

Abbildung 2: Softwarekarte für eine gegebene Fragestellung

Unsere Projektpartner nutzen bereits derartige graphische Darstellungen, die wir als *Softwarekarten* bezeichnen (siehe [LMW05]). Abbildung 2 zeigt beispielhaft eine Softwarekarte vom Typ Prozessunterstützungskarte, die die obige Fragestellung beantwortet und dabei die drei Informationsobjekttypen verwendet. Hierbei werden die Geschäftsprozesse auf der x-Achse angeordnet und die Lokationen auf der y-Achse platziert. Die Unterstützung eines Geschäftsprozesses durch ein Informationssystems an einem bestimmten Ort wird durch ein Rechteck an der entsprechenden Position visualisiert. Werden mehrere Prozesse unterstützt und/oder das System an mehreren Orten verwendet, so wird das Rechteck vergrößert bzw. dupliziert. Weitere Softwarekartentypen und deren Einsatzzweck beschreiben [MW04a] und [LMW05].

Um das Einsatzgebiet der Softwarekarte in Abbildung 2 zu erweitern und weitere Fragestellungen zu beantworten, verfolgen wir ein Schichten-Prinzip (vgl. [LMW05]), wel-

ches es ermöglicht, auf einem gleich bleibenden Kartengrund (gebildet durch *BusinessProcesses* und *Locations*) verschiedene relevante Aspekte darzustellen. Verbindungen zwischen den Systemen oder Betriebskosten könnten beispielsweise zusätzlich „aufgetragen“ werden, um weitere Fragestellungen zu beantworten. Insbesondere die Darstellung von Kennzahlen (vgl. [Be04]), wie sie beispielsweise ITIL [OG00] definiert, in bestehenden und verstandenen Darstellungen erhöht den Nutzen von Softwarekarten.

In unserem Projekt stellen wir derzeit einen entsprechenden Stakeholder- und Fragenkatalog zusammen, um ein geeignetes Informationsmodell zu entwickeln. Für die einzelnen Fragestellungen werden einzelne Sichten entwickelt, die entweder zu neuen Softwarekarten führen oder bestehende nutzen.

3 Einsatz des IEEE 1471 in der Softwarekartographie

Obige Beschreibung der Softwarekartographie erwähnt verschiedene Darstellungsformen, die für bestimmte Stakeholder relevante Aspekte des zu beschreibenden Systems (auf hinreichend hohem Abstraktionsniveau) abbilden. Eine ähnliche Vorgehensweise ist zentrales Leitbild des IEEE 1471, der Empfehlungen zur Beschreibung der Architektur von Software-intensiven Systemen abgibt [IE00]. IEEE 1471 hat die Unterstützung des *Dokumentierens*, *Explizierens* und *Kommunizierens* von Architekturen zum Ziel.

Im Gegensatz zu den verschiedenen Typen von Softwarekarten, die graphische Notationen zur Beschreibung von Anwendungslandschaften bereitstellen, adressiert der IEEE 1471 andere Probleme, die der Erstellung von eindeutigen, klaren und verständlichen Architekturbeschreibungen entgegenstehen: Eines der adressierten Probleme sind *unklare Begriffsverwendungen*.

Der Standard legt Definitionen zu Schlüsselkonzepten und Begriffen fest, soweit sich zu diesen eine Konsensbildung feststellen lässt [MEH01]. Im Folgenden stellen wir die Terminologie und die Konzepte des IEEE 1471 vor und verbinden diese mit der Softwarekartographie. Eine vergleichbare Vorgehensweise, bei der UML [OMG03] mit dem IEEE 1471 verbunden wird, verfolgt Hilliard in [Hi99].

Grundsätzlich befasst sich IEEE 1471 mit Software-intensiven Systemen, unter diese Bezeichnung fallen sämtliche Systeme, bei denen Software einen essentiellen Einfluss auf Design, Konstruktion, Einsatz (engl. *Deployment*) und Evolution des Systems in seiner Gesamtheit ausübt [IE00]. Damit umfasst IEEE 1471 auch Anwendungslandschaften. Van der Torre et al. entwickeln beispielsweise *Landscape Maps* für Unternehmensarchitekturen (*Enterprise Architectures*) unter Verwendung des IEEE 1471 [To04].

3.1 Terminologie und Konzepte

Die zwölf vom IEEE 1471 [IE00] in einem konzeptuellen Modell (siehe Abbildung 3) definierten Begriffe sind wie folgt:

Stakeholder sind nach dem IEEE 1471 Einzelpersonen, Teams oder Organisationen, die bestimmte *Concerns* (Interessen) am betrachteten System besitzen, analoges gilt auch bei der Darstellung von Anwendungslandschaften.

In IEEE 1471 kann eine *View* aus mehreren *Models* bestehen. Jedes dieser Modelle basiert auf im zugehörigen *Viewpoint* definierten Methoden inklusive Notationen, wobei sich ein *Viewpoint* zu einem *View* verhält wie eine Klasse zu einem Objekt. Zweck eines *Viewpoints* ist es, bei der Adressierung der ihm zugewiesenen *Concerns* unterstützend zu wirken. So kann beispielsweise eine *Management View* aus einem Organisationsmodell und einem Kostenmodell bestehen, wobei jedes Modell eine eigene Notation verwendet.

Die Softwarekartographie verwendet zur Beschreibung von Anwendungslandschaften die in Abschnitt 2 dargestellten, an die Kartographie angelehnten Notationen. Damit ist in der Softwarekartographie ein *Model* eine einzelne Softwarekarte, deren Notation in einem *Viewpoint* definiert wird. Eine *View* beschreibt damit (unter Nutzung eines oder mehrerer *Models*) die Architektur der Anwendungslandschaft aus einem bestimmten, durch die zum *Viewpoint* gehörigen *Concerns* festgelegten, Blickwinkel.

Als *Library Viewpoint* bezeichnet IEEE 1471 *Viewpoint*-Definitionen, die nicht im Rahmen der Erstellung der betrachteten Architekturbeschreibung entwickelt wurden. Solche *Viewpoint*-Definitionen können aus der Literatur (z.B. [Kr95]) stammen oder z.B. innerhalb einer Organisation als Standard vorgegeben sein. Im Rahmen einer bestimmten Architekturbeschreibung kann ein *Library Viewpoint* dann nach Anpassung an projektspezifische Gegebenheiten als *Viewpoint* zum Einsatz kommen. In der Softwarekartographie ist ein vordefinierter Softwarekartentyp (siehe [LMW05]), der einen bestimmten Kartengrund (z.B. Prozesse und Lokationen) und bestimmte Schichten (z.B. Informationssysteme und Betriebskosten) nutzt, um vorgegebene *Concerns* zu adressieren, ein *Library Viewpoint*.

Mit dem Begriff *System* bezeichnet IEEE 1471 das zu beschreibende Software-intensive System, welches ein Teil eines Systems, ein einzelnes System im Ganzen oder ein aggregiertes System (*system of systems* [IE00]) sein kann. In der Softwarekartographie ist das zu beschreibende System die Anwendungslandschaft oder Teile dieser.

Die *Architectural Description* setzt sich im IEEE 1471 aus den *Views* und *Models* zusammen. Die *Architectural Description* enthält somit alle für die *Stakeholder* relevanten Informationen zur Architektur.

Die *Rationale* beschreibt die Gründe, die zur Auswahl der Architektur geführt haben und den Zweck, den der Architekt mit seinen Entscheidungen verfolgt. Die Dokumentation von Sachverhalten bezüglich einer Architektur, die außerhalb der einzelnen *Views* stehen und für diese gemeinsam gelten, beschreibt auch [Cl03]. Dieses Vorgehen ist auch für Anwendungslandschaften sinnvoll, da es zu einem besseren Verständnis der Architektur beitragen kann und auch unnötiges Wiederaufleben von bereits geführten Diskussionen zu Architekturentscheidungen verhindert.

Die *Environment* des zu beschreibenden Systems wird vom IEEE 1471 als die Umwelt oder der Kontext, der einen Einfluss auf die Gestaltung des Systems ausübt, definiert.

Dies umfasst auch andere Systeme, mit denen das betrachtete System interagiert. Damit legt die Environment auch die Grenzen des betrachteten Systems fest. Für Anwendungslandschaften besteht die *Environment* aus allen Faktoren, die die Gestaltung der Anwendungslandschaft beeinflusst haben und beeinflussen. Zu diesen Einflussfaktoren zählen z.B. Organisationsstrukturen, Strategien oder technische Rahmenbedingungen.

Als *Mission* betrachtet IEEE 1471 den Verwendungszweck, den *Stakeholders* dem System zugedacht haben. Die grundlegende *Mission* der Anwendungslandschaft besteht in der Unterstützung von Primär- und Sekundärprozessen des Unternehmens.

3.2. Diskussion des IEEE 1471

Die Stärken des IEEE 1471 liegen vor allem in der Terminologie und den verwendeten Konzepten, die in Abschnitt 3.1 eingeführt wurden. Insbesondere das deutliche Hervorheben von *Stakeholder*, *Concerns* und *Views* stellt einen Wertbeitrag des IEEE 1471 dar. Architekturbeschreibungen (*Views*), die nicht auf die Interessen (*Concerns*) der an der Architektur interessierten Personen (*Stakeholders*) eingehen, sind wertlos.

Doch eben diese Terminologie kann auch irreführend sein, da beispielsweise das Konzept *Model* nicht mit der Verwendung im Model-View-Controller Entwurfsmuster übereinstimmt [Bu96]. Das *Model* im IEEE 1471 meint ein *Architectural Model*, welches bereits eine vom *Viewpoint* vorgegebene Notation zur Modellierung verwendet. Im Kontext von Anwendungslandschaften wird der Begriff *Model* häufiger für ein Informationsmodell [DMT99; LMW05] verwendet, welches die Objekttypen (Informationssysteme, Geschäftsprozesse, Kosten etc.) und deren Assoziationen definiert, aber noch keine Visualisierung festlegt. Auch Krcmar [Kr03] sieht die Visualisierung nicht notwendigerweise als einen Bestandteil eines Modells.

Des Weiteren stehen die Begriffe *Viewpoint* und *View* in einer 1:1 Beziehung, die auf den ersten Blick dem dazu im Standard als Beispiel aufgeführtem Verhältnis analog Klasse zu Objekt [IE00] widerspricht. Diese Assoziation wird mit einer Redundanzfreiheit begründet, da für eine bestimmte *Architectural Description* ein *Viewpoint* nur einen korrespondierenden *View* besitzt.

Wie z.B. in [St02] beschrieben, ist die Architektur eines Softwaresystems nicht statisch. Im Rahmen der Bestrebungen, der *Mission* gerecht zu werden, verändert sie sich ständig. Zur Beschreibung der Veränderung von Architekturdokumentationen bzw. Versionierung von Views liefert der IEEE 1471 keine Konzepte, was in zahlreichen Situationen ein Defizit des Standards darstellen kann.

Ein weiterer Kritikpunkt ist die verwendete Notation im konzeptuellen Diagramm des IEEE 1471. In der Notiz zum konzeptuellen Modell wird angemerkt, dass die Notation für Aggregationen von UML übernommen wurde, was nicht zwingend eine Übernahme der gesamten UML-Notation für das Diagramm darstellt. Dementsprechend ist eine adäquate Unterscheidung von Rollen- und Assoziationsnamen nicht zwingend, wäre aber wünschenswert.

4 Adaption des IEEE 1471 für die Softwarekartographie

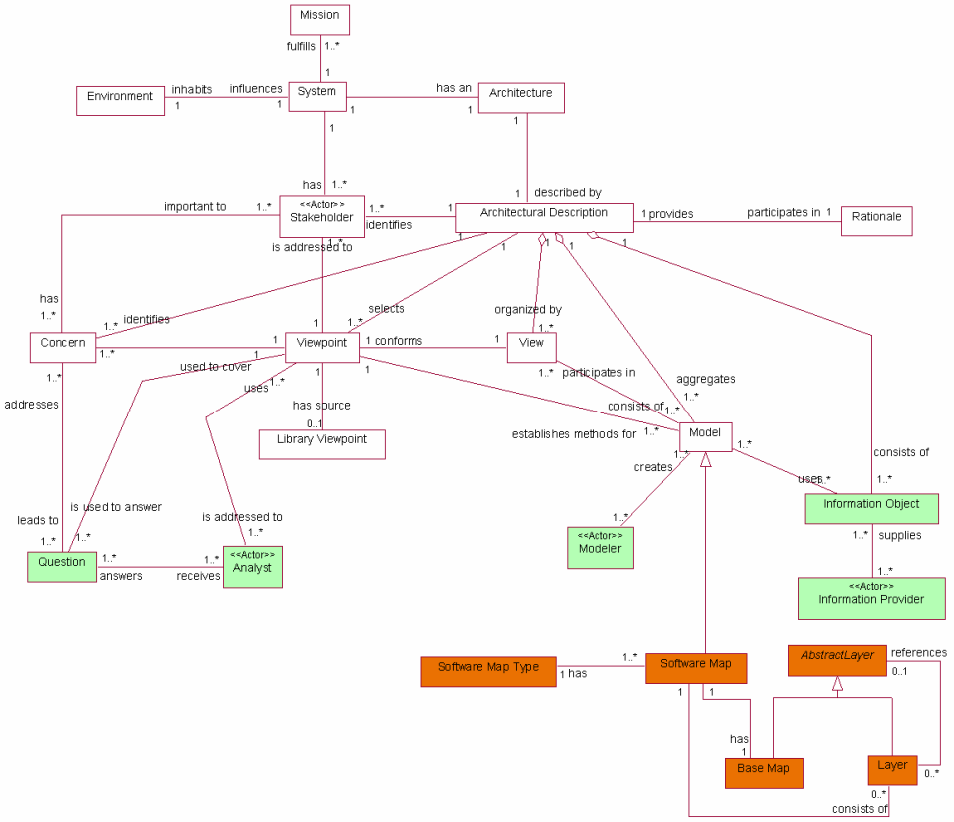

Abbildung 3: Verfeinerung des konzeptuellen Modells des IEEE 1471[1]

Bei dem Versuch, die Konzepte und Begriffe der Softwarekartographie in die Begriffswelt des IEEE 1471 einzuordnen, hat sich die Notwendigkeit von Anpassungen des konzeptuellen Modells ergeben. Diese verstehen sich nicht als Änderungsvorschlag für den Standard sondern als ein sehr stark auf IEEE 1471 aufbauendes konzeptuelles Modell, das der Beschreibung von Anwendungslandschaften auf abstrakter Ebene dient.

4.1 Besonderheiten beim Management von Anwendungslandschaften

Das ursprüngliche Hauptanwendungsgebiet von IEEE 1471 sind einzelne Softwaresysteme oder Produktlinien. Hier sollen aufbauend auf dem Anpassungsbedarf, den das *Conceptual Model* des IEEE 1471 bezüglich der Beschreibung von Anwendungsland-

[1] Das Klassendiagramm ist bei der Verwendung von Rollen- und Assoziationsnamen an die Darstellung im IEEE 1471 angelehnt.

schaften aufweist, die Verfeinerungen dargestellt und erläutert werden. Dazu werden Unterschiede zwischen der Betrachtung einzelner Softwaresysteme im Software Engineering und kompletten Anwendungslandschaften aufgezeigt.

Abbildung 3 zeigt unseren Vorschlag für die Anpassung des konzeptuellen Modells aus IEEE 1471. Der obere Teil, mit den Kästchen ohne farbliche Hinterlegung, zeigt das Modell des IEEE 1471, hinzugefügte Schlüsselbegriffe und Konzepte werden dort als graue (farbige) Kästchen dargestellt. Dabei stehen hellgraue (grüne) Kästchen für Verfeinerungen, die wir als notwendig für eine adäquate Unterstützung des Managements von Anwendungslandschaften im Allgemeinen ansehen. Optionale, auf die Softwarekartographie bezogene Verfeinerungen erscheinen als dunkelgraue (orange) Kästchen.

4.2 Fragestellungen und Analysten

Stakeholder sind nach IEEE 1471 Einzelpersonen, Teams oder Organisationen, die bestimmte Interessen am betrachteten System besitzen. Im *Requirements Engineering* ist es üblich, Stakeholder u.a. nach der Art ihrer Beziehung zum betrachteten System zu klassifizieren. Häufig findet sich dabei die Unterscheidung, ob ein Stakeholder sich direkt mit dem System befasst oder ob er lediglich vom Betrieb des Systems profitiert (z.B. *Functional Beneficiary* in [AR04] bzw. *Goal Stakeholder* in [PW01]).

Einzelne betriebliche Anwendungssysteme kommen oft in bestimmten Funktionsbereichen einer Organisation zum Einsatz (z.B. Administrations- und Dispositionssysteme wie Finanz-/Rechnungswesen, Personalbuchführung, Vertriebssysteme, Produktionsplanungssystem). In den entsprechenden Funktionsbereichen befindet sich dann oft ein Großteil der Nutzer, die „direkte“ Stakeholder darstellen. *Goal Stakeholder* sind dann zunächst die Mitarbeiter oder Führungskräfte in dieser Abteilung.

Bei der Betrachtung der Anwendungslandschaft als Gesamtheit aller betrieblichen Informationssysteme in einer Organisation stellt sich die Situation anders dar: Für Betrieb und Entwicklung der Anwendungslandschaft in der Gesamtheit ist kein einzelner Funktionsbereich verantwortlich, sondern eine Zentralstelle, der CIO oder die CIO-Abteilung. Ein Grund dafür liegt z.B. darin, dass der Nutzen aus Synergien, Querschnittsfunktionen, eingesparten Schnittstellen etc. oft nicht direkt einzelnen Fachbereichen zugerechnet werden kann. Damit besteht von Seiten dieser Fachbereiche geringes Interesse, Investitionen in anwendungsübergreifende Maßnahmen wie Architekturmanagement vorzunehmen. Ein sinnvolles Architekturmanagement, zu dem auch die Dokumentation der Anwendungslandschaft (und ihrer Architektur) gehört, fällt in den Aufgabenbereich der Unternehmensleitung bzw. der CIO-Abteilung [Wi04 Ga04].

Nicht in allen Fällen werden Mitglieder des Topmanagements selbst *Views* analysieren, um ihre *Concerns* bezüglich der Anwendungslandschaft zu evaluieren. Stattdessen werden sie Mitarbeiter oder auch ein externes Evaluationsteam, wie z.B. in [CKK02] empfohlen, einsetzen, das sich mit den entsprechenden *Concerns* bezüglich der Anwendungslandschaft beschäftigt. Da *Views* den Kenntnisstand und andere Eigenschaften ihrer Nutzer berücksichtigen müssen, die sich oft von den *Stakeholders*, die durch das in der *View* zu untersuchende Problem betroffen sind, unterscheiden (vgl. auch [An04]), ist

es hier wichtig, zwischen dem Nutzer der *View* und dem *Stakeholder*, dessen *Concerns* adressiert werden sollen, zu unterscheiden.

Diese Unterscheidung führt zu einer Verfeinerung des IEEE 1471, der nur den *Stakeholder* in seinem konzeptuellen Modell integriert hat. In Abbildung 3 haben wir neben dem *Stakeholder* und seinen *Concerns* zusätzlich *Questions* und *Analysts* eingeführt, wobei zusätzlich der *Stakeholder* und *Analyst* als *Akteure* annotiert werden.

Hierdurch wird eine Unterscheidung der abstrakten *Concerns* (z.B.: Ein bestimmter Prozess soll durch ausfallsichere Anwendungen unterstützt werden) und den abgeleiteten *Questions* (siehe Fragestellungen in Abschnitt 2) ermöglicht. Diese *Questions* (z.B. „Welche Abhängigkeiten und Zyklen bei Batch-Läufen existieren zwischen den Informationssystemen?“) ermöglichen es dem *Analyst* verschiedene *Views* aufzubauen oder auszuwählen, die einzelne Fragestellungen (*Questions*) beantworten, um einzelne oder mehrere *Concerns* anzusprechen.

4.3 Modellierer und Informationslieferanten

Die Erstellung von Architekturdokumentationen zu einem bestehenden System ist eine wichtige Problemstellung im Rahmen von *Reengineering-Projekten* ([KWC98]). Der Aufbau einer Architekturbeschreibung bzw. eines Architekturmodells wird dort als Aufgabe des Softwarearchitekten oder einer vergleichbaren Rolle angesehen. Dies wird stark erleichtert, wenn z.B. die kompletten Quelltexte inklusive Dokumentation für ein Softwaresystem in einem zentralen Repository (z.B. CVS) zugänglich sind. Bei einer Anwendungslandschaft muss hingegen zunächst sichergestellt werden, dass beschreibende Daten (in ausreichender Qualität) in einem zentralen Repository bereitstehen.

Damit stellt die Gewinnung der in den Architekturmodellen abzubildenden Information eine der zentralen Herausforderung und einen bedeutenden Kostenfaktor bei der Erstellung der Dokumentation einer Anwendungslandschaft dar. Dieser Kostenfaktor ist sowohl in der Theorie [Cl03] als auch in der Praxis [Ha04] bekannt und muss bei Entscheidungen zur Dokumentation der Architektur berücksichtigt werden. Problematisch in diesem Zusammenhang erweist sich, dass die Kosten zur Datenerhebung u.U. nicht an denselben Stellen anfallen, an denen der Nutzen aus der Architekturbewertung (Adressierung der *Concerns* von *Stakeholdern*) auftritt.

Um diese Sachverhalte abzubilden, führt unsere Verfeinerung des IEEE 1471 unter anderem zwei Rollen ein: *Modeler* und *Information Provider*. Der *Modeler* ist dabei für die Erstellung der konkreten *Models*, die als Bestandteil einer oder mehrerer *Views* die Architektur der Anwendungslandschaft (*System*) abbilden, zuständig. Dazu nutzt er die Methoden (darunter auch die Modellnotationen), die der entsprechende *Viewpoint* für das *Model* definiert. Zum Aufbau von Modellen nutzt der *Modeler Information Objects*, die von verschiedenen *Information Providers* bereitgestellt werden. Diese *Information Objects* bilden auch einen Bestandteil der Architekturbeschreibung.

4.4 Kommunikation der Akteure

Das Zusammenwirken der einzelnen Akteure[2] (*Stakeholder*, *Analyst*, *Modeler* und *Information Provider*) stellt Abbildung 4 dar: Bei *Stakeholders* treten bestimmte *Concerns* auf (Nachricht 1 in Abbildung 4), die teilweise auch direkt als *Questions* formuliert werden können (2). *Concerns* und ggf. *Questions* werden den *Analyst* weitergegeben (3). Dieser verfeinert (4), die vom *Stakeholder* erhaltenen *Questions*, und führt ggf. neue *Questions* ein (5), um ein bestimmtes *Concern* besser zu erfassen.

Auf Basis der *Questions* wählt der *Analyst* einen zur Bearbeitung geeigneten *Viewpoint* aus (6), um die im *Viewpoint* definierten *Models* vom *Modeler* erstellen zu lassen (7). Hierdurch ergibt sich beim *Modeler* ein Informationsbedarf, er fordert den *Information Provider* auf, die entsprechenden *Information Objects* bereitzustellen (8). Dieser kommt der Aufforderung nach (9).

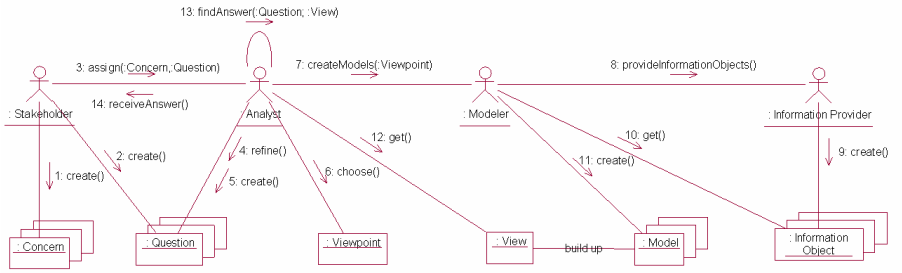

Abbildung 4: Kommunikationsdiagramm zur Erstellung und Nutzung von Softwarekarten

Der *Modeler* greift auf die *Information Objects* zu (10), um die benötigten *Models* zu erstellen (11), die in ihrer Gesamtheit die zum *Viewpoint* passende *View* bilden. Der *Analyst* nutzt diese *View* (12) zur Beantwortung der an ihn gerichteten *Questions* (13). Abschließend werden die Antworten an den *Stakeholder* geleitet (14).

4.5 Softwarekarten und Schichten

Um den IEEE 1471 für die im Forschungsprojekt Softwarekartographie entstehenden bzw. entstandenen Darstellungen nutzbar zu machen, wurden zusätzlich Softwarekarten (*Software Maps)*, als eine Spezialisierung des *Models*, in die Verfeinerung aufgenommen. Der Softwarekartentyp (*Software Map Type*) unterscheidet Softwarekarten nach ihrem Aufbau und Einsatzgebiet [LMW05].

Softwarekarten bestehen aus mehreren Schichten (*Layers;* vgl. Abschnitt 2), die auf einem Kartengrund (*Base Map*) aufgetragen werden. Dieser Kartengrund stellt einen Spezialfall einer Schicht dar und ist für eine Softwarekarte obligatorisch, darüber liegende Schichten referenzieren entweder den Kartengrund oder andere Schichten.

---

[2] Akteure sind als Rollen in einer Organisation zu verstehen, wobei mehrere Rollen von der gleichen Person wahrgenommen werden können.

Das Schichtenprinzip ermöglicht es, unterschiedliche Fragestellungen unter Wiederverwendung eines bestehenden Kartengrundes zu beantworten. Durch einen gleich bleibenden Kartengrund ist für den Analysten ein erhöhter Wiedererkennungswert gegeben. Um die Übersichtlichkeit zu wahren, ist es je nach Fragestellung möglich, Schichten ein- und auszublenden.

5. Zusammenfassung und Ausblick

In diesem Beitrag haben wir gezeigt, dass der IEEE 1471 einen geeigneten terminologischen Rahmen für die Softwarekartographie bildet. Aufgrund des hochgradig arbeitsteiligen, verzahnt und kontinuierlich ablaufenden Architekturmanagementprozesses halten wir jedoch eine Verfeinerung des IEEE 1471 für notwendig. Für unsere Projektpartner, die teilweise bereits den IEEE 1471 nutzen, ist mit diesem Beitrag auch ein Leitfaden entstanden, der beim Einsatz des IEEE 1471 zum Management von Anwendungslandschaften unterstützt. Insbesondere die Verwendung von klaren Begrifflichkeiten und die Ausrichtung von *Views* an *Stakeholders* und deren *Concerns* haben sich bereits im Projekt bewährt. Die Anwendung des Leitfadens zeigt erste Erfolge, im weiteren Verlauf des Projektes Softwarekartographie werden wir die Methodik weiter ausbauen und validieren.

Literaturverzeichnis

[An04] Andrade, J. et al.: A Methodological Framework for Viewpoint-Oriented Conceptual Modeling. IEEE Transactions on Software Engineering 30(5), S. 282-294, 2004.

[AR04] Alexander, I.; Robertson, S.: Understanding Project Sociology by Modeling Stakeholders. IEEE Software 21 (1), S. 23-27, 2004.

[Be04] Beyer, N.: Kennzahlen zur Beschreibung von Anwendungslandschaften und ihre Visualisierung auf Softwarekarten. Bachelor's Thesis, Fakultät für Informatik. München: Technische Universität München, 2004.

[Bu96] Buschmann, F. et al.: Pattern-Oriented Software Architecture, Volume 1: A System of Patterns. New York, Wiley & Sons, 1996. ISBN 0-471-95869-7.

[CKK02] Clements, P.; Kazman, R.; Klein, M.: Evaluating Software Architectures. 1. Auflage, Addison-Wesley, 2002. ISBN 0-201-70482-X.

[Cl03] Clements, P. et al.: Documenting Software Architectures: Views and Beyond. 1. Auflage, Boston, Addison-Wesley, 2003. ISBN 0-201-70372-6.

[DMT99] DMTF: Common Information Model (CIM) Specification - Version 2.2. Distributed Management Task Force, Inc. (DMTF), 1999. http://www.dmtf.org/standards/documents/CIM/DSP0004.pdf [abgerufen 2004-10-20].

[Ga04] Gaertner, W.: Ansatz für eine erfolgreiche Enterprise Architecture im Bereich Global Banking Division/Global Transaction Banking IT and Operations der Deutschen Bank. Wirtschaftsinformatik 46 (4), S. 311-313, 2004.

[Ha04] Hanschke, I.: IT-Governance: Bestandteile und deren Zusammenspiel. Vortrag, Workshop „IT-Management in der Praxis“, München, 13 Oktober 2004.

[Hi99] Hilliard, M.: Using the UML for Architectural Description. In: "UML" '99 - The Unified Modeling Language: Beyond the Standard, Second International Conference, Fort Collins, CO, USA, 1999. ISSN 0302-9743.

[HGM02] Günter Hake, Dietmar Grünreich und Liqui Meng: Kartographie. 8. Auflage, Berlin, New York: de Gruyter, 2002. ISBN 3-11-016404-3.

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

[Kr03] Krcmar, H.: Informationsmanagement. 3. Auflage, Berlin, Heidelberg: Springer, 2003.

[Kr95] Kruchten, P.: The 4+1 View Model Software Architecture. IEEE Software 12 (11), S. 42-60, 1995

[KWC98] Kazman, R.; Woods, S.; Carrière, S.: Requirements for Integrating Software Architecture and Reengineering Models: CORUM II. In: Working Conference on Reverse Engineering (WCRE'98), 1998.

[LMW05] Lankes, J.; Matthes, F.; Wittenburg, A.: Softwarekartographie: Systematische Darstellung von Anwendungslandschaften. In: 7. Internationale Tagung Wirtschaftsinformatik 2005, Bamberg, Germany, 2005 (noch nicht erschienen).

[MEH01] Maier, M.; Emery, D.; Hilliard, R.: Software Architecture: Introducing IEEE Standard 1471. IEEE Computer, S. 107-109, 2001.

[MW04a] Matthes, F.; Wittenburg, A.: Softwarekartographie: Visualisierung von Anwendungslandschaften und ihrer Schnittstellen. In: Informatik 2004 - Informatik verbindet, 34. Jahrestagung der GI, Ulm, Deutschland, 2004. ISBN 3-88579-380-6.

[MW04b] Matthes, F.; Wittenburg, A.: Softwarekarten zur Visualisierung von Anwendungslandschaften und ihrer Aspekte. Technische Universität München, Lehrstuhl für Informatik 19 (sebis), München, Deutschland, TB0402, 2004.

[OG00] Office of Government Commerce (OGC): ITIL - Service Delivery. Norwich, UK: The Stationery Office, 2000. ISBN 0-11-330017-4.

[OMG03] OMG: Unified Modeling Language Specification, Version 1.5. OMG, 2003.

[PW01] Preiss, O.; Wegmann, A.: Stakeholder Discovery and Classification Based on Systems Science Principles. In: APAQS 2001, Hong Kong, 2001.

[St02] Starke, G.: Effektive Software-Architekturen. Ein praktischer Leitfaden. München, Wien: Hanser, 2003. ISBN 3-44-621998-6

[To04] van der Torre, L. et al.: Landscape Maps for Enterprise Architectures. Information Centre of Telematica Instituut, Technischer Bericht TI/RS/2004/016, 2004.

[Wi04] Winter, Robert: Architektur braucht Management. Wirtschaftsinformatik 46 (4), S. 317-319, 2004.