Technische Universität München

Fakultät für Informatik

Bachelorarbeit in Wirtschaftsinformatik

Retrospektive Analyse der kundenspezifischen Anpassungen eines EA Management Werkzeugs

Ex-post analysis of customizations of an EA management tool

Markus Sebastian Bauer

Prüfer:	Prof. Dr. rer. nat. Florian Matthes
Betreuer:	Dipl. Inf. Christian M. Schweda
Abgabedatum:	15.09.2010

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

München, den …… ……

Abstract

Information models build the fundamental infrastructure for documenting, i.e. describing, enterprise architectures (EAs). As every enterprise is likely to have its own understanding of EA reflected in a distinct terminology and enterprise-specific information demands, up to now no 'standard information model' for describing EAs has emerged. Nevertheless, different EA management approaches as well as tools for supporting EA management bring along a predefined information model that should be adapted to the specific needs of the using enterprise.

The field of information model adaptation has yet not been explored in-depth, most notably only little is known on the adaptation techniques that the information model users would expect to have. In addition, such knowledge is mostly kept secret by tool vendors and consultancies that help enterprises in establishing an organization-specific EA management. The situation in information model adaptation is far from the comprehensive documentation of techniques in the field of reference model adaptation.

In this thesis, we approach the topic of techniques for information model adaptation in a twofold way. Firstly, we explore the state-of-the-art in literature and, as literature is currently scarce, analyze the adaptation techniques used by selected users of an open-source EA management tool. Secondly, we revisit literature and techniques from bordering fields and use the findings to elicit requirements for possible adaptation techniques from the tool's user group determining possible adaptation techniques. In addition, we map these requirements to the tool resulting in possible solutions for their implementation. Thereby, the thesis serves as a collection of improvements in the area of modeling. We also propose exemplary approaches for the further development of additional adaptation mechanisms.

Abkürzungsverzeichnis

IS	Informationssystem
IT	Information Technology
EA	Enterprise-Architecture
EAM	Enterprise-Architecture Management
SOA	Serviceorientierte Architektur

Abbildungsverzeichnis

Tabellenverzeichnis

Inhalt

1 Motivation und Einführung

Unternehmen sind heute mehr denn je gezwungen, agil und in angemessener Weise auf sich häufig ändernde Bedingungen des sozialen und wirtschaftlichen Umfelds zu reagieren (vgl. [Fi07]). Der sich globalisierende Markt stellt seit Jahren immer mehr Unternehmen vor die Situation, mit neuen, teilweise interkontinentalen Wettbewerbern in Konkurrenz treten zu müssen. Diese sich verschärfende Wettbewerbssituation setzt Unternehmen zunehmend unter Druck, die eigene Entwicklung, sowohl im Hinblick auf die Optimierung bestehender Prozesse und Produkte, als auch die Weiterentwicklung des Produkt- und Dienstleistungsportfolios an sich, kontinuierlich voranzutreiben (vgl. [Su09]). Ein Bereich, in dem eine derartige Weiterentwicklung in den letzten Jahrzehnten sehr deutlich zu beobachten ist, besteht in der noch immer stark zunehmenden IT-Unterstützung der Geschäftsprozesse. Dabei werden nach [Kr10] nicht nur bereits etablierte Prozesse durch den Einsatz von Anwendungssystemen optimiert, vielmehr werden neue Geschäftsprozesse oft erst durch die Entwicklungen in diesem Umfeld ermöglicht. Da jedoch bei einer Vielzahl moderner Geschäftsprozesse nicht nur einzelne Anwendungssysteme beteiligt sind, entstehen durch diese Entwicklung immer komplexer werdende Zusammenhänge zwischen Geschäftsprozessen und deren IT-Unterstützung (vgl. [Ha10]). Um in dieser Komplexität richtige Entscheidungen treffen zu können, ist es nötig die Zusammenhänge in ihrer Gesamtheit zu betrachten. Ein Ansatz den immer mehr Unternehmen verfolgen, besteht im Instrumentarium des Unternehmensarchitekturmanagement (engl. „Enterprise Architecture Management“ - EAM), welches nach [Bu08] einen gesamtheitlichen Blick auf die beschriebenen Zusammenhänge ermöglicht.

Obwohl die Relevanz des EAMs sowohl in der Praxis, als auch in der Forschung stetig wächst (vgl. [Bu08]), konnte sich bis heute kein EAM-Ansatz („EAM-Framework“) bzw. keine, diesen Ansätzen zugrunde liegende, allgemein gültige Definition dieses Begriffs durchsetzen. Ein EAM-Framework definiert dabei nach [Mi10] Prozesse sowie Ordnungsrahmen und enthält Checklisten, die man für die Erarbeitung einer Unternehmensarchitektur benötigt. Darüber hinaus beschreiben solche Frameworks EAM-typische Fragestellungen, sowie die für die Beantwortung dieser Fragestellungen benötigte Methoden und Prozesse.

Das Fehlen einer allgemein anerkannten EAM-Definition bedingt die „Unschärfe“ des Konzepts, welche auch gleichzeitig zur Problemstellung dieser Arbeit führt. Die Vielzahl an bekannten EAM-Frameworks bilden zwar Hilfestellungen für Unternehmen, die sich entscheiden EAM einzuführen, doch gibt es meist unternehmensspezifische IT- oder Geschäftskonzepte, die nach Auffassung des jeweiligen Unternehmens, beim Management der Unternehmensarchitektur zusätzliche Beachtung finden sollten. Im Gegensatz zu unternehmensspezifischen Erweiterungen, sind auch Reduzierungen des vom Framework beschriebenen Ansatzes denkbar. Hierbei werden von EAM-Frameworks empfohlene bzw. vorgesehene Konzepte

vom jeweiligen Unternehmen als irrelevant erachtet und daher nicht in das unternehmensspezifische EAM bzw. die darin enthaltene Modellierung der Unternehmensarchitektur mit übernommen.

Diese unternehmensspezifischen Ausgestaltungen des Begriffs der Unternehmensarchitektur, stellen Entwickler eines EAM-Frameworks bzw. eines EAM-Werkzeugs vor eine große Herausforderung. Zum einen sollten die Unternehmen beim Aufbau und der Pflege ihrer Unternehmensarchitektur, sowie bei der Beantwortung von EAM-Fragestellungen bestmöglich unterstützt werden. Gleichzeitig sollte dem Unternehmen aber auch größtmögliche Flexibilität zur Abbildung unternehmensspezifischer EA-Konzepte gewährleistet werden. Diese Flexibilität spiegelt sich bei EAM-Werkzeugen in dem der Unternehmensarchitektur zugrundeliegenden Informationsmodell wieder. Die hierzu verfolgten Ansätze, mit denen verschiedene Werkzeughersteller versuchen, die entgegenläufigen Anforderungen an eine größtmögliche Flexibilität, sowie an eine gleichzeitig größtmögliche Unterstützung der Einführung bzw. dem Betrieb des EAMs umzusetzen, reichen dabei vom Ausschluss jeglicher Anpassungen, bis hin zu eigenständigen Erstellung eines EA-Informationsmodells durch das jeweilige Unternehmen (vgl. [Bu08]).

In dieser Arbeit soll nun am Beispiel eines ausgewählten EAM-Werkzeugs, welches bei der Gewichtung der soeben beschriebenen Anforderungen eine Kompromisslösung darstellt, untersucht werden, welche Techniken zur Anpassung des Informationsmodells des Werkzeugs zur Verfügung gestellt werden, wie diese in der Praxis bei der unternehmensspezifischen Modellierung der Unternehmensarchitekturen zur Anwendung kommen und welche Probleme dabei auftreten können. Das Ziel dieser Untersuchungen liegt dabei in der Identifikation von Anforderungen an die Anpassbarkeit des Informationsmodells, welche durch die Adaptionsmöglichkeiten des EAM-Werkzeugs zum Untersuchungszeitpunkt nicht unterstützt werden. Im Anschluss werden Lösungen erarbeitet, welche die Anpassbarkeit des Werkzeugs entsprechend dieser Anforderungen erweitern.

1.1 Vorgehensweise

Um das im vorigen Abschnitt beschriebene Ziel zu erreichen, bestehen die Hauptuntersuchungsgegenstände der im Rahmen dieser Arbeit durchgeführten Untersuchungen, aus den durchgeführten Informationsmodellanpassungen einzelner Unternehmen. Um diese Analysen zu ermöglichen, werden zuvor das EAM-Framework, welches dem untersuchten Werkzeug – iteraplan – zugrunde liegt, sowie das Werkzeug selbst, detailliert vorgestellt. Besondere Beachtung gilt dabei den allgemein möglichen, sowie im Speziellen, den von iteraplan bereitgestellten Adaptionsmöglichkeiten, die es Anwendern ermöglichen, ihre unternehmensspezifischen EA-Konzepte auf EAM-Werkzeuge bzw. auf iteraplan abzubilden.

Nachdem die möglichen Adaptionen sowie allgemein denkbare Anforderungen an die Adaptionsfähigkeit von EA-Informationsmodellen bekannt sind, kann mit der eigentlichen Analyse begonnen werden. Diese besteht im Vergleich von EA-Informationsmodellen einzelner Unternehmen mit dem Informationsmodell von iteraplan, sowie der Betrachtung von Informationsmodellanpassungen des iteraplan-Informationsmodells durch ausgewählte Organisationen. Das Ziel dieser Untersuchung liegt dabei im Vergleich der jeweiligen Modellierungsabsichten mit den tatsächlichen Modellierungsergebnissen, sowie der damit einhergehenden Identifikation etwaiger Diskrepanzen. Im Anschluss an die Identifikation dieser Diskrepanzen ist zu prüfen, ob diese auf das Fehlen eines geeigneten Adaptionsmechanismus hindeuten. Die daraus gewonnenen Erkenntnisse werden danach u.U. in zusätzliche Anforderungen an die Adaptionsfähigkeit des Werkzeugs übertragen und anschließend im Hinblick auf ihre Umsetzbarkeit hin geprüft.

1.2 Gliederung

In Kapitel 2 wird der Kontext dieser Arbeit, das Enterprise Architecture Management vorgestellt. Neben der allgemeinen EAM-Einführung, gibt der Abschnitt auch einen kurzen Überblick über verschiedene Ansätze der EAM-Werkzeugunterstützung. Zur Schaffung einer dieser Arbeit zugrundeliegenden Terminologie werden wichtige EAM-Konzepte eingeführt, beschrieben bzw. definiert.

In Kapitel 3 werden anschließend das für die Untersuchungen in dieser Arbeit ausgewählte EAM-Werkzeug iteraplan, sowie dessen zugrunde liegender EAM-Ansatz des Herstellers detailliert vorgestellt. Besonderes Interesse gilt dabei zusätzlich dem Erweiterungsmechanismus, welcher die unterstützten Adaptionsmöglichkeiten des Informationsmodells des Werkzeugs implementiert.

Die Adaptionsmöglichkeiten bilden in Kapitel 4 den wichtigsten Untersuchungsgegenstand der eigentlichen Analyse dieser Arbeit. Vor der Untersuchung konkreter Beispiele, werden an dieser Stelle allgemeine Anforderungen an die Anpassbarkeit von Informationsmodellen beschrieben, um daraus die Untersuchungsmethodik abzuleiten. Zusätzlich werden diese allgemeinen Anforderungen auf ihre Unterstützung durch das Werkzeug geprüft. Die Durchführung der im Anschluss vorgestellten Analyse untergliedert sich in den Vergleich von EA-Informationsmodellen einzelner Unternehmen mit dem Informationsmodell des EAM-Werkzeugs, sowie die Betrachtung in iteraplan durchgeführter Informationsmodellanpassungen ausgewählter Unternehmen.

Im Anschluss an diese Untersuchung werden in Kapitel 5 die gefundenen Erkenntnisse in Änderungsanforderungen an die vom Werkzeug bereitgestellten Anpassungsmöglichkeiten übertragen, bzw. Lösungsansätze erarbeitet, welche die identifizierten Änderungsanforderungen umsetzen.

Abschließend werden in Kapitel **Fehler! Verweisquelle konnte nicht gefunden werden.** die rgebnisse dieser Arbeit zusammengefasst. Zusätzlich werden die in Kapitel 5 beschriebenen Erweiterungen des Werkezugs auf ihre Umsetzbarkeit bzw. deren Vereinbarkeit mit Ansatz, der dem untersuchten EAM-Werkzeug zugrunde liegt, geprüft.

2 Enterprise Architecture Management

Die seit Jahrzehnten stark wachsende IT-Durchdringung in Unternehmen führt immer häufiger zu unüberschaubaren IT-Landschaften aus hunderten von Anwendungssystemen, Schnittstellen und Hardwarekomponenten, die zusammen hochkomplexe Strukturen bilden (vgl. Abbildung 1). Um diese Komplexität beherrschen zu können, setzen Unternehmen vermehrt auf EAM. Dieser Ansatz verspricht heute eine ganzheitliche Sicht auf das Unternehmen, seine internen Strukturen sowie deren jeweilige Beziehungen zur IT-Landschaft.

Die Geschichte des EAM reicht dabei zurück bis in die Mitte der 1980er Jahre, als John A. Zachmann einen ersten entsprechenden Ansatz vorstellte ([Za87]). Seither entwickelten und verbreiteten sich mehrere EA-Ansätze („Frameworks“), wobei jeder spezifische Auffassungen von EAM vertritt. EA- bzw. EAM-Frameworks bilden nach [Hi10] Werkzeuge, die Prozesse und Ordnungsrahmen definieren, und gleichzeitig Checklisten enthalten, die man zur Erarbeitung einer Unternehmensarchitektur benötigt. Die bekanntesten Vertreter derartiger EA-Frameworks sind dabei nach [Ha10]:

- *The Open Group Architecture Framework* (TOGAF, siehe [TO09])
- *US Federal Enterprise Architecture Framework* (FEAF, siehe [Sk04])
- *Department of Defense Architecture Framework (DoDAF, siehe [Do04a], und [Do04b])*
- *Extended Enterprise Architecture Framework* (E2AF, siehe [Sk04])
- *Integrated Architecture Framework* (IAF, siehe [En08])

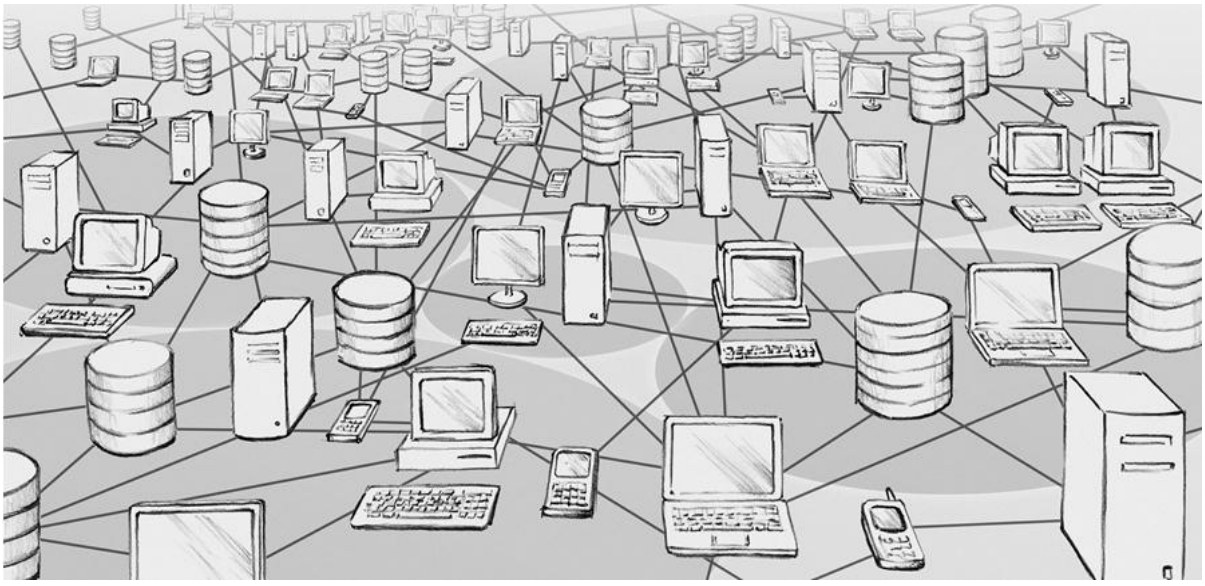

Abbildung 1 – Komplexität der IT-Landschaft
(Quelle: [Ha10])

Von diesen Frameworks konnte sich jedoch bis heute keiner als „Beste Lösung“ von den anderen abheben bzw. durchsetzen. Da jedem dieser Ansätze gleichzeitig auch eine jeweils ei-

gene, allgemeine Definition von EAM zugrunde liegt, existiert bis heute ebenfalls keine alleingültige Definition für dieses Konzept. Dies führt direkt zur Problemstellung dieser Arbeit: Die abweichenden Auffassungen von EAM allgemein, bzw. von einzelnen EAM-Teilaspekten verschiedener EAM-Steakholder. Der Begriff EAM-Stakeholder fasst dabei diejenigen Institutionen zusammen, welche ein wirtschaftliches, informatives oder sonstiges Interesse an EAM besitzen, und steht somit für Organisationen und Personen, die an der Entwicklung eines EAM-Frameworks bzw. eines EAM-Werkzeugs beteiligt, oder von der Einführung einer EAM-Lösung betroffen sind.

Um trotz des Fehlens einer allgemein anerkannten Definition des EAM-Ansatzes (vgl. [Pr09]) ein gemeinsames Grundverständnis für den weiteren Verlauf dieser Arbeit zu schaffen, wird die Definition von [Ma08] herangezogen:

> *„Enterprise architecture management is a continuous and iterative process controlling and improving the existing and planned IT support for an organization. The process not only considers the information technology (IT) of the enterprise, also business processes, business goals, strategies etc. are considered in order to build a holistic and integrated view on the enterprise.*
>
> *Goal is a common vision regarding the status quo of business and IT as well as of opportunities and problems arising from these fields, used as a basis for a continually aligned steering of IT and business."*

Demnach beschreibt EAM allgemein einen Prozess, der versucht sowohl die derzeitige, als auch die zukünftige IT-Unterstützung des Unternehmens zu planen bzw. zu steuern. Diese Definition wurde gewählt, da sich die in ihr getroffenen Hauptaussagen auch in einer Reihe anderer Definitionen finden lassen. Einen Einblick in den Diskussionsbedarf bezüglich einer allgemein anerkannten Definition des Begriffs liefert [Mal10].

Die allgemeinen Definitionen enthalten jedoch keine Aussagen über die Abläufe dieser Planung bzw. Steuerung. Die Beschreibung dieser Ebene, d.h. die Beschreibung der jeweiligen Ziele, die durch das EAM erreicht werden sollen, sowie die Methoden die zur Erreichung dieser Ziele verwendet werden können, wird dabei von den einzelnen EAM-Frameworks übernommen. Diese ergänzen also die abstrakte Definition des EAM-Ansatzes, um konkrete Beschreibungen der zur Erreichung des übergeordneten EAM-Vorhabens nötigen Teilschritte.

2.1 EAM – Patterns

Die oben beschriebenen EA-Frameworks definieren neben Methoden zur Planung und Steuerung der Unternehmensarchitektur, auch die dafür benötigte Daten bzw. Datentypen („Elementtypen“). Die Vielzahl an verbreiteten EAM-Frameworks sowie die jeweils eigenen Terminologien dieser Ansätze, haben zu einem Überangebot an Begriffen und Ausprägungen ähnlicher bzw. gleicher EA-Konzepte geführt. Als EA-Konzepte gelten dabei Elemente wie Methoden, Ziele, Datentypen o.Ä., die durch das jeweilige Framework beschrieben werden.

Ein Ansatz, der versucht die in den verschiedenen Frameworks definierten und in der Praxis verwendeten EA-Konzepte generisch zu beschreiben, und gleichzeitig als Einführungs- sowie Erweiterungshilfe für Unternehmen, die EAM betreiben wollen, zu dienen, wird von [Bu08] vorgestellt. Hierzu wurden 43 wichtige Ziele (*Concerns*) identifiziert, die durch das EAM erreicht werden sollten. Diese können dabei durch 20 beschriebene Methoden (*M-Patterns*) verfolgt werden. Die Ziele und ihre zugeordneten Methoden werden zur Vereinfachung in sieben Themenkomplexe gegliedert:

1. ***Technology Homogenity*** beschreibt Methoden, die zur Analyse und Steuerung der technologischen und architekturbezogenen Homogenität der *Applikationslandschaft* (*application landscape*) dienen.
2. ***Business Process*** behandelt auf hohem Abstraktionsniveau die Zusammenhänge zwischen *Applikationen* (*business applications*) und *Geschäftsprozessen* (*business processes*).
3. ***Application Landscape Planning*** beschreibt die Analyse sowie die Planung der Entwicklung der *Applikationslandschaft* im Hinblick auf Ist-, Plan- und Soll-Zustand.
4. ***Support of Business Processes*** beinhaltet Methoden zur Analyse der IT-Unterstützung von Geschäftsprozessen.
5. ***Project Portfolio Management*** beinhaltet Methoden zur Steuerung des Portfolios von Projekten, die zur Änderung der *Applikationslandschaft* führen.
6. ***Infrastructure Management*** analysiert die *Technische Infrastruktur*, auf der die *Applikationslandschaft* betrieben wird, und den Einfluss, den Änderungen der Infrastruktur auf die IT-Unterstützung hat.
7. ***Interface, Business Object, and Service Management*** fasst die Methoden zusammen die zur Analyse der *Services* im Kontext Serviceorientierter Architekturen („SOA“) dienen.

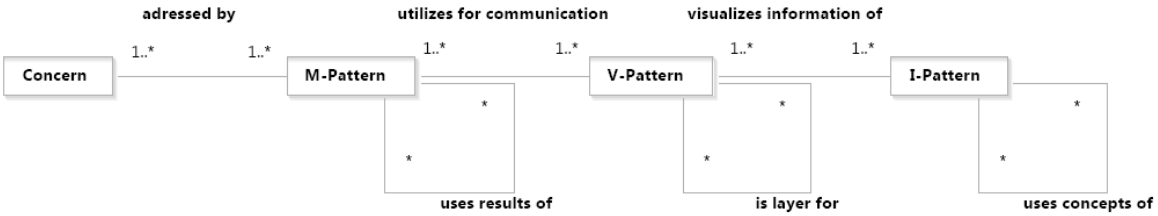

Abbildung 2 – Vereinfachtes EAM-Pattern Informationsmodell (Quelle: [Bu08])

Darauf aufbauend beschreibt [Bu08] 53 EAM typische Visualisierungen (*V-Patterns*) die bestimmte *Concerns* bzw. *M-Patterns* adressieren, sowie 47 Informationsmodellfragmente (*I-Patterns*), die die Daten für die entsprechenden *V- und M-Patterns* abbilden. Die *I-Patterns* definieren dabei neben den EAM-typischen Element- bzw. Datentypen, auch deren Beziehungen untereinander. Diese bilden somit die Bausteine für das EA-Informationsmodell des jeweiligen Unternehmens.

Demnach kann die Gesamtheit aller EAM-Konzepte in vier Schichten untergliedert werden (vgl. Klassen in Abbildung 2). Die Anwendung des Ansatzes besteht dabei nach [Bu08] in der unternehmensspezifischen Selektion relevanter Ziele (*Concerns*) und Methoden (*M-Patterns*), wodurch indirekt auch gleichzeitig die verbundenen *V-* und *I-Patterns* bestimmt werden, was wiederum den unternehmensspezifischen EAM-Ansatz vervollständigt.

2.2 EAM – Werkzeuge

Das bereits beschriebene, teilweise stark abweichende Verständnis von EA-Konzepten einzelner Unternehmen, sowie die stark wachsende Anzahl der EAM-Stakeholder, führen zu einer Vielzahl an Anforderungen an die EAM-Werkzeuge. Eine vollständige Abdeckung all dieser Anforderungen ist dabei nach [Pe07b] nahezu unmöglich. Dies stellt EAM-Werkzeughersteller vor das Problem, die Verfolgung möglichst vieler EAM-Ziele (vgl. *Concerns*) zu unterstützen, und gleichzeitig den anwendenden Unternehmen, die größtmögliche Flexibilität bei der Selektion der für sie wichtigen *Concerns* zu bieten. Wie in Abbildung 2 dargestellt, betrifft eine solche Selektion bestimmter *Concerns* jedoch auch die Modellierung der entsprechenden Informationsmodellfragmente. Somit kann der Anspruch an eine größtmögliche Flexibilität von EAM-Werkzeugen, auf die Anforderung an eine flexible Modellierung des EA-Informationsmodells übertragen werden. Die Wichtigkeit der Anpassungsmöglichkeiten eines EAM-Werkzeugs, wird von Forrester - einem der führenden Unternehmen im Bereich EAM-Research - bei der Untersuchung der unterstützten Anwendungsfälle einzelner EAM-Werkzeuge betont. Dies geschieht sowohl 2007 als auch 2009 durch die höchste Gewichtung der Kategorie der Informationsmodellierung (vgl. „*Modeling*“ [Pe07a], [Pe09a]).

Die unterschiedliche Bewertung der diametralen Anforderungen an eine umfangreiche Unterstützung möglichst vieler *Concerns*, bei einer gleichzeitig größtmöglichen Flexibilität durch die verschiedenen Werkzeughersteller, führt zu einer Kategorisierung der den Werkzeugen zugrunde liegenden Ansätzen: In Anlehnung an [Ma08] können EAM Werkzeuge bezüglich der Dimension „Flexibility vs. Guidance“ in die drei Kategorien *metamodellorientiert*, *methodenbasiert* und *prozessorientiert* unterteilt werden, wobei die Abgrenzung dabei nicht immer scharf vorgenommen werden kann.

- ***Metamodellorientierte*** Werkzeuge verfolgen den Ansatz der größtmöglichen Flexibilität bezüglich des Informationsmodells. Dies bedeutet, dass das Werkzeug im Initialzustand zwar ein Informationsmodell enthält, das Unternehmen dieses Modell jedoch verändern kann. So können unternehmensspezifische EA-Konzepte entwickelt, und gleichzeitig Lösungen erarbeitet werden, die diese Konzepte modellieren. Hierzu werden die Benutzer meist in Form von grafischen Editoren bei der Modellierung unterstützt.

  Der Vorteil eines solchen Ansatzes liegt sicherlich in der großen Flexibilität bei der Ausgestaltung des Werkzeugs. So lassen sich mit diesen Werkzeugen auch „unübliche“ EAM-Ziele verfolgen, zu deren Beantwortung stark unternehmensspezifische Fragestellungen notwendig sind. Der Nachteil besteht jedoch in der Einführungsphase, in der das spezifische Informationsmodell erarbeitet werden muss. Außerdem kann die Erfahrung der Hersteller derartiger Werkzeuge nur bedingt in die kundenspezifischen Definitionen einzelner Informationsmodelle fließen. Die Qualität der

Modellierung hängt dabei stark von der Qualifikation des Modellierers ab. Ein weiterer Nachteil von *metamodellorientierten* EAM Werkzeugen spiegelt sich in den generischen Auswertungsmöglichkeiten wieder. Bedingt durch die Flexibilität des zugrundeliegenden Informationsmodells kann der Hersteller keine auf das Modell zugeschnittenen EAM-typische Auswertungs- und Visualisierungsmöglichkeiten zur Verfügung stellen. Somit müssen auch hier die Unternehmen in Eigenverantwortung Abfragen und grafische Auswertungen definieren, oder auf die kostenpflichtige Erfahrung des Herstellers zurückgreifen.

- Im Kontrast zu den *metamodellorientierten* Werkzeugen stehen die ***prozessorientierten*** Werkzeuge. Diese bieten, verglichen mit den zuvor Beschriebenen, keine bzw. nur stark eingeschränkte Möglichkeiten zur Anpassungen des Informationsmodells, bei einer gleichzeitig größtmöglichen Unterstützung der vom Werkzeug beschriebenen EAM-Prozesse. Das bedeutet, dass ein *prozessorientiertes* Werkzeug stets mit einem vordefinierten Informationsmodell installiert wird, bei dem nur geringfügige Anpassungen vorgenommen werden können. Ergänzend zum Informationsmodell definiert das Werkzeug Benutzerrollen, so dass Benutzer nur bestimmte – in ihrer Verantwortung liegende – Schritte des gesamten EAM-Prozesses vornehmen können. Dies lässt sich beispielhaft an einem Werkzeug darstellen, in dem es nicht direkt möglich ist Elemente vom Typ „Projekt“ anzulegen. Der vordefinierte Prozess schreibt hingegen vor, dass zuerst ein „Projektvorschlag“ angelegt wird, der nach Beurteilung durch einen anderen Benutzer mit entsprechenden Berechtigungen in ein „Projekt“ umgewandelt werden kann. Diese strikte Vorgabe von Anwendungsfällen gilt auch bei den Auswertungsmöglichkeiten dieser Werkzeuge.
  Die umfassende und exakte Definition aller zugelassenen Anwendungsfälle erleichtert die schnelle Einführung von EAM bzw. die EAM-Werkzeugunterstützung ohne lange Vorlaufzeit und verkürzt somit die Zeit bis zur Inbetriebnahme des Werkzeugs. Demgegenüber steht die Kritik an den strikt definierten Prozessen, besonders wenn das Werkzeug in einem Unternehmen mit bereits bestehenden EAM-Strukturen eingeführt werden soll.

- Als Mittelweg können die EAM-Werkzeuge bezeichnet werden, die einen ***methodenbasierten*** Ansatz verfolgen. Diese kombinieren in begrenztem Ausmaß die Anpassbarkeit *metamodellorientierter* mit der Prozessführung, wie sie bei *prozessorientierten* Werkzeugen zu finden ist. *Methodenbasierte* Werkzeuge liefern im Initialzustand ein vordefiniertes Informationsmodell, welches auf den Erfahrungen des Herstellers aufbaut und in der Regel Möglichkeiten zur Anpassungen bereitstellt. Gleichzeitig gibt es meist eine Reihe vordefinierter Auswertungen die bei Bedarf an die jeweiligen Anforderungen angepasst werden können. Dies verkürzt, wie auch bei

den *prozessorientierten* Werkzeugen, die Einführungszeit des Werkzeugs, ohne dabei die Flexibilität des Unternehmens zu stark einzuschränken.

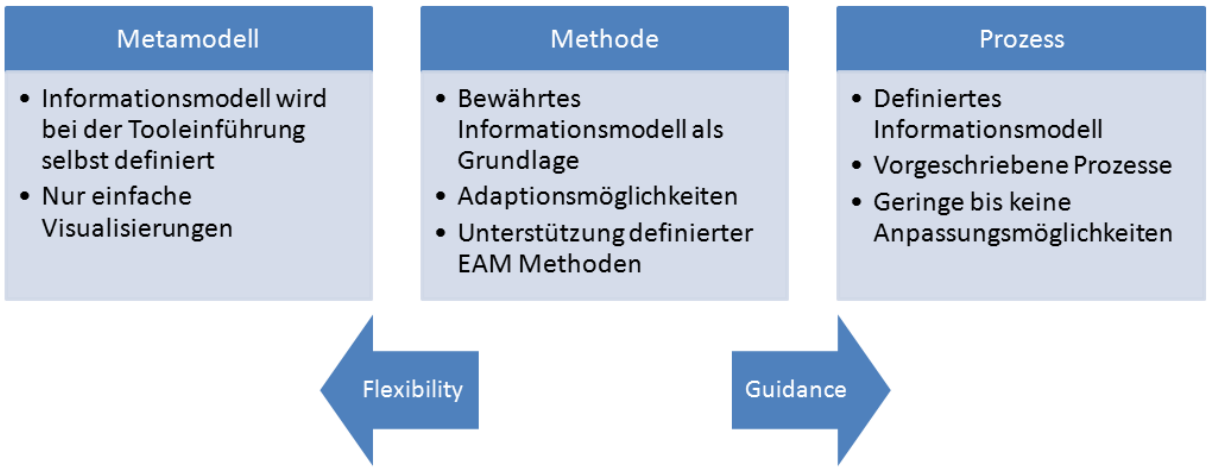

Abbildung 3 - EAM Toolklassifikation
(Quelle: Eigene Grafik, in Anlehnung an [Bu08])

Im weiteren Verlauf dieser Arbeit wird nun das für die Untersuchungen ausgewählte EAM-Werkzeug iteraplan, im Hinblick auf dessen Anpassbarkeit an die unternehmensspezifischen Anforderungen näher betrachtet. iteraplan gehört nach [Pe09b] zu den wichtigsten Open Source EAM-Werkzeugen, und kann in Anlehnung an die von [Ma08] beschriebene Kategorisierung, den *methodenbasierten* Werkzeugen zugeordnet werden. Die iteraplan zugrundeliegende EAM-Methodik wird dabei durch das von der iteratec GmbH, dem Hersteller des EAM-Werkzeugs, entwickelten EAM-Framework beschrieben (vgl. [Ha10]).

3 iteraplan

Wie bereits erwähnt ist iteraplan ein Open Source EAM-Werkzeug, das auf dem von der iteratec GmbH entwickelten EA-Framework basiert. Technologisch ist iteraplan eine JavaEE Web-Anwendung, die durch die Frameworks Spring und Hibernate sowie auf Basis einer relationalen Datenbank in einer klassischen Dreischicht-Architektur mit Präsentations-, Geschäftslogik- und Datenhaltungsschichten realisiert wurde.

Die erste Version von iteraplan entstand 2004 und wurde anfangs ausschließlich im Rahmen von Kundenprojekten der iteratec GmbH eingesetzt. Die Folgeversionen 1.0 bis 1.8 wurden damals als Closed Source Projekte weiterentwickelt. Anfang 2007 entschied sich die iteratec GmbH, iteraplan künftig als Open Source Projekt weiterzuentwickeln und stellte den Quellcode von iteraplan 2.0 öffentlich unter der AGPL Lizenz auf SourceForge zur Verfügung (siehe [So10]). Seitdem wird iteraplan stetig von einem Entwicklerteam bei iteratec weiterentwickelt. Die in dieser Arbeit untersuchte Version (iteraplan 2.6) wurde im April 2010 veröffentlicht.

3.1 iteratec-EA-Framework

Die iteratec GmbH hat ihre Erfahrung aus einer Vielzahl von IT-Beratungsprojekten genutzt, um daraus ein eigenes EA-Framework zu entwickeln („*Best-Practice Enterprise Architecture*“). Mit diesem Ansatz will iteratec eine praxiserprobte Komplettlösung anbieten, die es Unternehmen sowohl ermöglicht, EAM ohne lange Vorbereitungszeit einzuführen, als auch an die unternehmensspezifischen Anforderungen anzupassen. Hierzu wird von [Ha10] eine Vorgehensweise vorgestellt, die Unternehmen einen einfachen Einstieg in das komplexe Thema des EAMs ermöglichen soll. Das Framework beschreibt ein methodisches Vorgehen zur Erarbeitung und Einführung eines unternehmensspezifischen EAM-Ansatzes. Den Ausganspunkt eines solchen Ansatzes bilden dabei definierte EAM-Ziele, welche durch beschriebene Methoden und Visualisierungen erreicht werden können. Die Vorgabe dieser Ziele, sowie der zugehörigen Methoden bzw. Visualisierungen führt zusätzlich zur Definition eines Architekturmodells (vgl. Abbildung 4). Dieses Architekturmodell definiert EAM-Elementtypen („*Buildingblocktypen*“), Architekturebenen sowie die Beziehungen zwischen diesen Ebenen. Die definierten Architekturebenen sind dabei:

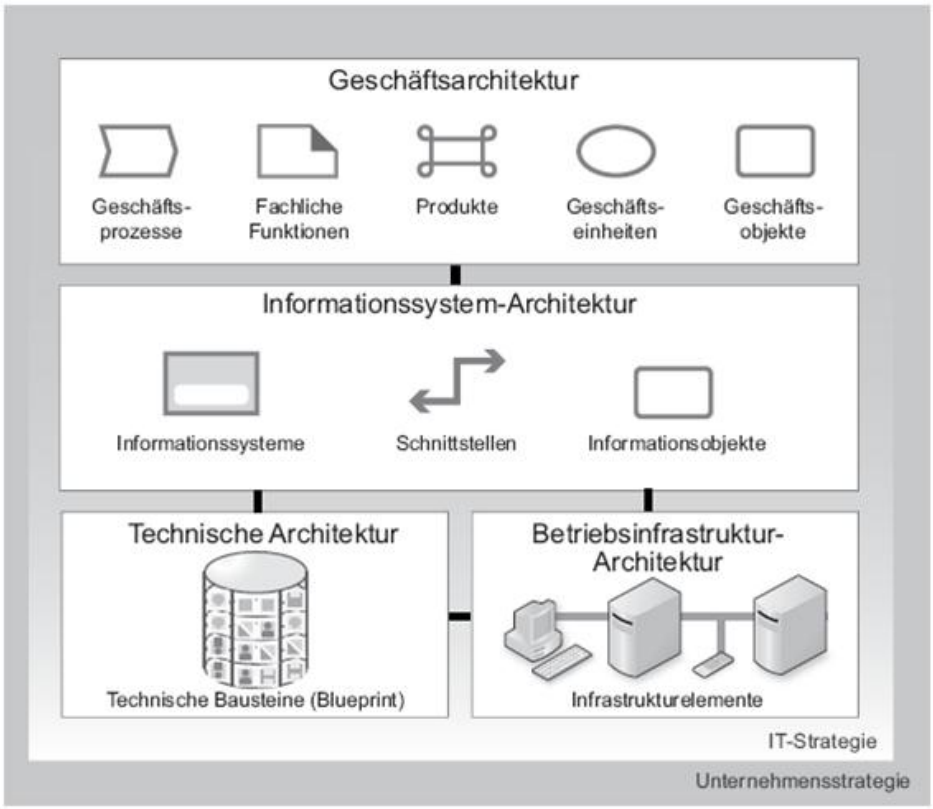

Abbildung 4 – EA-Framework iteratec
(Quelle: [Ha10])

- Die ***Geschäftsarchitektur*** beschreibt die grundlegenden Geschäftsstrukturen des Unternehmens und beinhaltet die dafür wichtigen Elementtypen *Geschäftsprozess*, *Fachliche Funktion*, *Produkt*, *Geschäftseinheit* und *Geschäftsobjekt*. Diese Architekturebene entspricht der in Abschnitt 2.1 beschriebenen Kategorie des *Business Process*. Sie beschreibt mit ihrer Verbindung zur *Informationssystem-Architektur* neben den Geschäftsstrukturen auch deren IT-Unterstützung.
- Die ***Informationssystem-Architektur*** beinhaltet die Elementtypen *Informationssysteme*, *Schnittstellen* und *Informationsobjekte*. Diese dienen zur Beschreibung der Anwendungslandschaft der Unternehmen. Die *Informationssystem-Architektur* bildet das Bindeglied zwischen der Geschäftsarchitektur und der technischen sowie der *Betriebsinfrastruktur-Architektur*. Die Verbindung zur *Geschäftsarchitektur* soll dabei aufzeigen, welchen Beitrag die IT-Unterstützung zur Wertschöpfung des Unternehmens leistet und wie dieser Beitrag erbracht wird. Die technische Implementierung der einzelnen *Informationssysteme* und *Schnittstellen* wird durch die Zuordnung zu entsprechenden Elementen der *Technischen Architektur* abgebildet. Die Zuordnung zu Elementen der *Betriebsinfrastruktur-Architektur* bildet dagegen ab, auf welchen Hardwarekomponenten die jeweiligen *Informationssysteme* und *Schnittstellen* betrieben werden. Somit beschreibt diese Architekturebene Elementtypen, die die *Concerns* der

von [Bu08] beschriebenen Kategorie des *Interface, Business Object, and Service Managements* reflektieren.

- Die ***Technische Architektur*** dient zur Definition unternehmensspezifischer technischer Standards für die Entwicklung von *Informationssystemen, Schnittstellen* und *Infrastrukturelementen*. Hierzu kann sie Referenzarchitekturen, Templates, fremdentwickelte IT-Produkte, Werkzeuge für Softwareerstellung und Systemmanagement sowie Frameworks auf den Elementtyp *Technischer Baustein* abbilden, und diese Elementen der *Betriebsinfrastruktur-Architektur* bzw. der *Informationssystem-Architektur* zuordnen. Somit adressiert diese Architekturebene die *Concerns* der Kategorie *Technology Homogenity*.
- Die ***Betriebsinfrastruktur-Architektur*** bildet auf einem hohen Abstraktionsniveau die Infrastruktur ab, auf der die *Informationssysteme* des Unternehmens betrieben werden. Hierzu werden die *Infrastrukturelemente*, über die Verbindung zur *Informationssystem-Architektur*, den entsprechenden *Informationssystemen* und *Schnittstellen* zugeordnet. Gleichzeitig werden technische Standards durch die Zuordnung von Elementen der angrenzenden *Technischen Architektur* abgebildet. Diese Architekturebene bildet also die Informationen ab, die den *Concerns* der Kategorie *Infrastructure Management* zuzuordnen sind.

Die beiden umgebenden Strategieebenen der ***IT-*** und der ***Unternehmensstrategie*** dienen zusätzlich zur Betonung der Wichtigkeit einer ganzheitlichen Betrachtung des Unternehmens, sowie zur Verknüpfung der übergeordneten Unternehmensziele mit den Entwicklungen der IT. Das Ziel dieses Frameworks besteht nach [Ha10] in der Bereitstellung eines Frameworks, der mit geringem Aufwand eingeführt und an die unternehmensspezifischen Anforderungen angepasst werden kann. Die Anpassung des Frameworks wird dabei, ähnlich wie der von [Bu08] beschriebene Ansatz, durch die Selektion von Zielen, Methoden, Visualisierungen sowie deren zugehörige Informationsmodellfragmente vorgenommen. Diese Konzepte werden jedoch nach der von [Ha10] beschriebenen Terminologie, den Kategorien „Analyse-Muster“, „Planungs-Muster“ und „Modellierungsrichtlinien“ zugeordnet, und somit nicht in verschiedene Schichten gegliedert, sondern in jeweiligen Funktionskomplexen zusammengefasst.

Aufbauend auf diesem EAM-Framework, hat die iteratec GmbH iteraplan entwickelt, und ermöglicht somit die Werkzeugunterstützung der von [Ha10] beschriebenen EAM-Methodik.

3.2 Informationsmodell

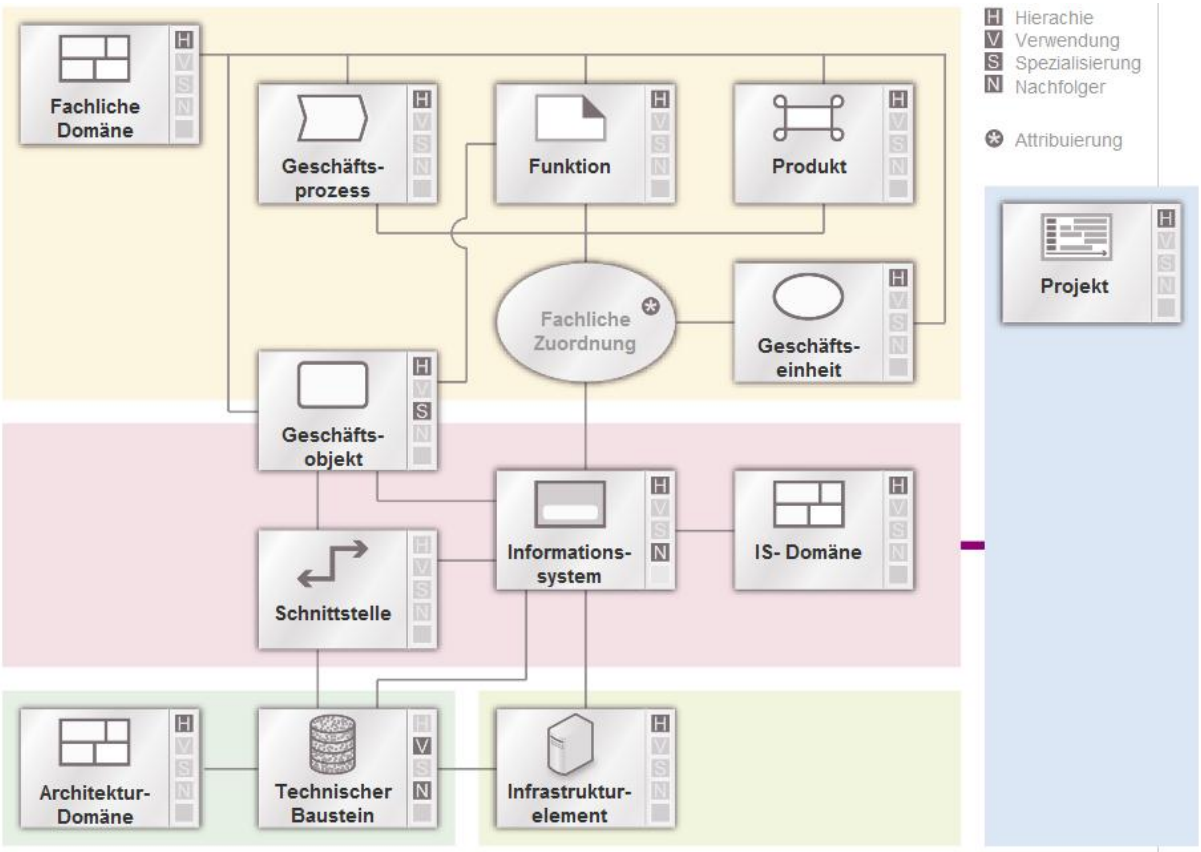

Abbildung 5 – iteraplan-Informationsmodell
(Quelle: [it10])

Um den von [Ha10] beschriebenen Anspruch, an eine sofortige Verwendbarkeit des vorgestellten EA-Frameworks auf das darauf aufbauende iteraplan zu übertragen, setzt die iteratec GmbH bei der Bewertung der Anforderungen an eine größtmögliche Abdeckung der EAM-typischen Ziele, auf die Unterstützung bestimmter EAM-Methoden, deren entsprechende Visualisierungen sowie den zugrunde liegenden Informationsmodellfragmenten. Dies führt zu dem in Abbildung 5 dargestellten Informationsmodell der *Buildingblocktypen* von iteraplan, das gleichzeitig den Ausganspunkt für die Entwicklung der Funktionalität des Werkzeugs bildet.

In Anlehnung an das von [Ha10] beschriebene Architekturmodell, ist das Informationsmodell ebenfalls in fünf Ebenen untergliedert, wobei die Ebenen der *Betriebsinfrastruktur-*, *Geschäfts-*, *Informationssystem-* sowie der *Technischen-Architektur*, eine direkte Entsprechung im Informationsmodell besitzen. Dagegen werden die *IT-* und *Unternehmensstrategie* durch eine Ebene repräsentiert, welche ausschließlich den *Buildingblocktyp Projekt* enthält. Dieser dient zur Planung und Steuerung der IT-Landschaft, entsprechend der vom Unternehmen verfolgten Strategie.

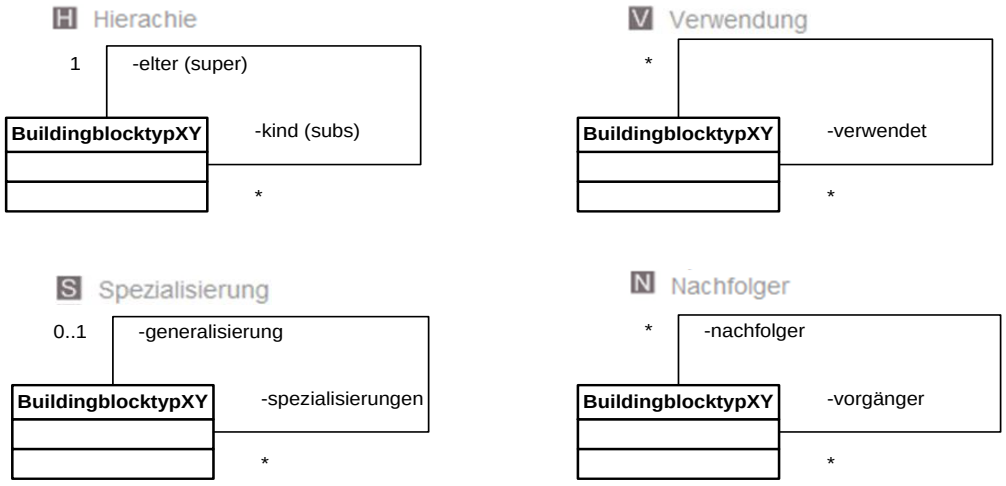

Abbildung 6 – iteraplan Modellierungsmuster

Um die *Informationssystem-Architektur* weiter zu untergliedern wurde neben den im iteratec EA-Framework vorgestellten *Buildingblocktypen* zusätzlich *Informationssystemdomänen* eingeführt. Die Kategorisierung der *Technischen Architektur* wird auf ähnliche Weise durch die Definition von *Architekturdomänen* ermöglicht. Der letzte *Buildingblocktyp*, der zusätzlich zu den bereits im Abschnitt 3.1 vorgestellten Elementen definiert wurde, besteht in der *Fachlichen Zuordnung*. Dieser Elementtyp beschreibt jedoch keinen weiteren *Buildingblocktypen* im eigentlichen Sinn. Vielmehr setzen sich Elemente dieses Typs aus jeweils einer Instanz der *Buildingblocktypen Informationssystem, Geschäftsprozess, Geschäftseinheit* sowie *Produkt* zusammen. Somit bilden diese die Information ab, welche *Informationssysteme* an welcher Stelle des Wertschöpfungsprozesses des Unternehmens unterstützend tätig sind (vgl. *Support of Business Processes*, Abschnitt 2.1).

Buildingblocktyp	Klasse
Architekturdomäne	*ArchitecturalDomain*
Fachliche Domäne	*BusinessDomain*
Fachliche Funktion	*BusinessFunction*
Geschäftseinheit	*BusinessUnit*
Geschäftsobjekt	*BusinessObject*
Geschäftsprozess	*BusinessProcess*
Informationssystemdomäne	*InformationSystemDomain*
Informationssystem	*InformationSystem, InformationSystemRelease*
Infrastrukturelement	*InfrastructureElement*
Produkt	*Product*
Projekt	*Project*
Schnittstelle	*InformationSystemInterface*
Technischer Baustein	*TechnicalComponent, TechnicalComponentRelease*

Tabelle 1 – Modellklassen der Buildingblocktypen

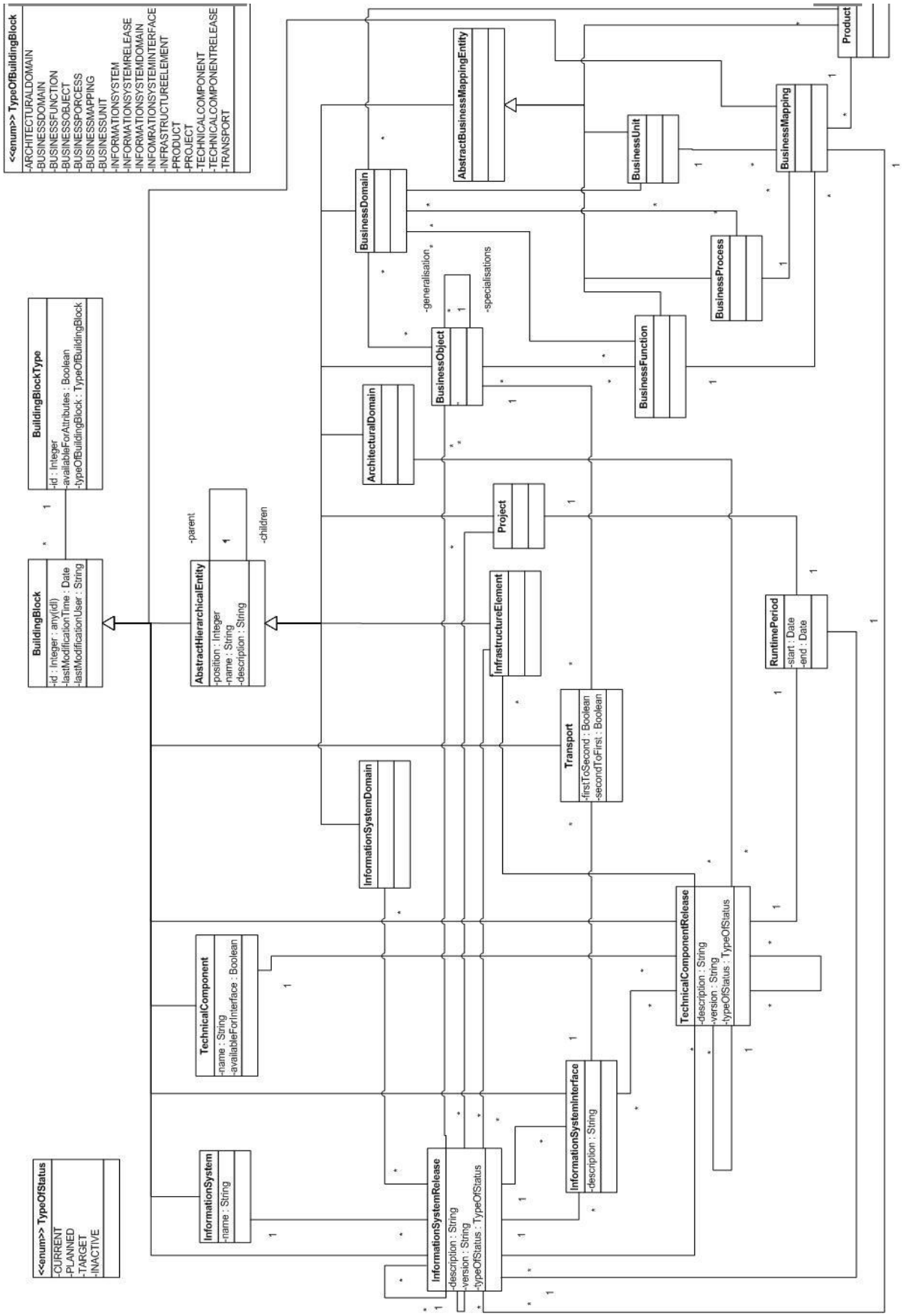

Abbildung 7 - iteraplan Datenmodell der Buildingblocktypen

Eine weitere Differenz zwischen dem iteratec EA-Framework und dem Informationsmodell von iteraplan liegt in der gemeinsamen Abbildung von *Geschäftsobjekten* und *Informationsobjekten* des Frameworks auf den *Buildingblocktyp Geschäftsobjekt* des Informationsmodells. Dies spiegelt die Anordnung der Klasse über die Grenzen der beiden Ebenen hinweg wieder. Diese Kombination von *Geschäftsobjekten* und *Informationsobjekten* stützt sich darauf, dass bei Unternehmen mit einer hohen IT-Durchdringung die *Informationsobjekte* der *Informationssystem-Architektur* im Idealfall eine Entsprechung in *Geschäftsobjekten* der *Geschäftsarchitektur* besitzen. Der letzte Schritt zur Überführung des Architekturmodells in das Informationsmodell besteht aus der Auflösung der Beziehungen zwischen den einzelnen Architekturebenen, in Beziehungen zwischen den entsprechenden *Buildingblocktypen*. Um die Beziehungen, wie sie in Abbildung 5 abgebildet sind, verständlich zu machen und gleichzeitig ein gemeinsames Verständnis der einzelnen *Buildingblocktypen* für den weiteren Verlauf dieser Arbeit zu schaffen, werden diese kurz vorgestellt. Zusätzlich werden einige typische Beispiele für Instanzen des jeweiligen Typs angebracht:

- ***Architekturdomänen*** dienen zur Strukturierung des Standardisierungskatalogs der technischen Bebauung (*Technischen Architektur*). Typische Beispiele für *Architekturdomänen* sind „Datenbanken“, „Application Server“, usw.
- ***Fachliche Domänen*** bilden fachliche Einteilungen zur Strukturierung der *Buildingblocktypen* der *Geschäftsarchitektur*. Ein Beispiel für eine *Fachliche Domäne* stellt der „Verkauf“ bzw. „Vertrieb“ dar.
- ***Fachliche Funktionen*** bilden in sich abgeschlossene sowie zusammenhängende fachliche Tätigkeiten, wie das Anlegen eines Kunden, ab.
- ***Geschäftseinheiten*** sind die strukturellen Einheiten der Organisationsstruktur wie z.B. Abteilungen, Standorte oder Werke eines Unternehmens.
- ***Geschäftsobjekte*** stellen einen fachlichen Begriff für abstrakte bzw. konkrete Objekte dar, die in engem Zusammenhang zur Geschäftstätigkeit des Unternehmens stehen. (z.B. „Kunde“, „Rechnung“, usw.).
- ***Geschäftsprozesse*** sind Folgen von logisch zusammenhängenden Aktivitäten, die für das Unternehmen einen Beitrag zur Wertschöpfung leisten, einen definierten Anfang und ein definiertes Ende haben und in der Regel wiederholt durchgeführt werden (z.B. „Kundenakquise“).
- ***Informationssystemdomänen*** fassen nach einem oder mehreren Kriterien verschiedene *Informationssysteme* zusammen. Sie werden häufig verwendet, um die Anwendungslandschaft sowie Verantwortlichkeiten für das EAM aufzuteilen (z.B. „Operative Systeme“, „Management-Informationssysteme“, usw.).
- ***Informationssysteme***, „Applikationen“ bzw. „Anwendungen“ bilden den wichtigsten *Buildingblocktypen* zur Beschreibung der *Informationssystem-Architektur*. Sie bilden

sinnvolle Elemente, die vom Anwender als fachliche Einheit angesehen werden, und setzten sich für gewöhnlich aus einer Präsentations-, einer Logik- sowie einer Datenhaltungsschicht zusammen (z.B. „SAP R/3“).

- ***Infrastrukturelemente*** umfassen die Hardware- und Netzwerkeinheiten, auf denen die Elemente der *Informationssystem-Architektur* betrieben werden wie z.B. „Oracle DB-Server“.
- ***Produkte*** sind die materiellen und immateriellen Ergebnisse des Wertschöpfungsprozesses der Organisation.
- ***Projekte*** sind IT-Vorhaben, Programme oder Maßnahmen, die als eigenständiger Umfang geplant und gesteuert werden. Als typische Beispiele hierfür gelten Migrationen von *Informationssystemen* oder *Infrastrukturelementen*.
- ***Schnittstellen*** definieren Abhängigkeiten zwischen zwei *Informationssystemen* im Kontext des Informationsflusses. Sie beschreiben hierzu die Richtung sowie die *Geschäftsobjekte* der Informationsübertragung.
- ***Technische Bausteine*** liefern die Informationen zur technischen Realisierung von *Informationssystemen* und *Schnittstellen* (z.B. „Java“, „MySQL“, usw.).

Zusätzlich zu den Beziehungen zwischen den verschiedenen *Buildingblocktypen,* bildet das Informationsmodell von iteraplan auch Autoassoziationen einiger Elemente ab. Aus Übersichtlichkeitsgründen wurden hierzu Modellierungsmuster verwendet, die in Abbildung 6 dargestellt sind. So werden die Autoassoziationen *Hierarchie, Spezialisierung, Nachfolger* sowie *Verwendung* durch ihre jeweiligen Anfangsbuchstaben im Informationsmodell repräsentiert und dadurch den entsprechenden Elementen zugeordnet. Ein *Buildingblocktyp* implementiert das entsprechende Modellierungsmuster dabei immer genau dann, wenn der jeweilige Anfangsbuchstabe innerhalb der Visualisierung des *Buildingblocktyps* nicht ausgegraut dargestellt ist. Das Implementieren der *Hierarchie* erfordert dabei das Vorhandensein einer Instanz, welche als Wurzelelement dieser Hierarchie dient.

Abbildung 7 zeigt das Ergebnis der Übertragung des in Abbildung 5 dargestellten Informationsmodells, auf ein UML-Kassendiagramm der *Buildingblocktypen*. Tabelle 1 ordnet den *Buildingblocktypen* die entsprechenden Modellklassen zu. Das UML-Klassendiagramm definiert das vordefinierte Datenbankschema der Buildingblockklassen von iteraplan (Version 2.6), welches zwar nicht verändert werden, dafür ermöglicht es eine schnelle Einführung des Werkzeugs und verringert den, der Einführung vorausgehenden Planungsaufwand. Um dennoch Unternehmen die Möglichkeit zu geben, das Informationsmodell bzw. das Werkzeug adäquat an die spezifischen Anforderungen anzupassen, stellt iteraplan einen Erweiterungsmechanismus zur Verfügung, der nachfolgend detailliert vorgestellt wird.

3.3 Attributmechanismus

Um die Anpassbarkeit von iteraplan an die unternehmensspezifischen EAM-Anforderungen zu gewährleisten, wurde von der iteratec GmbH der sogenannte „Attributmechanismus“ entwickelt (vgl. [it10]). Dieser ermöglicht es, zusätzliche *Merkmale* für ausgewählte *Buildingblocktypen* zu definieren. Hierzu können von Anwendern mit entsprechenden Berechtigungen, *Aufzählungs-*, *Freitext-*, *Datums-*, *Verantwortlichkeits-*, sowie *Zahlmerkmale* definiert werden, und für bestimmte *Buildingblocktypen* freigeschaltet werden. Zusätzlich zu den freigeschalteten *Buildingblocktypen*, kann für jeden *Merkmalstyp* festgelegt werden, ob es sich dabei um ein Pflichtmerkmal handelt. Dies hat zur Folge, dass von iteraplan definierte Konsistenzchecks, auf Instanzen hinweisen, bei denen zum Zeitpunkt der Überprüfung kein Wert für das Pflichtmerkmal eingepflegt wurde. Ferner kann bei *Aufzählungs-* und *Verantwortlichkeitsmerkmale* die Kardinalität definiert werden, indem die obere Schranke als eins bzw. unbeschränkt definiert wird. Die verbleibenden *Merkmalstypen* besitzen dagegen ausschließlich eine obere Schranke von 1. Für *Zahlenmerkmale* können Einheit, sowie die untere und obere Schranke der Wertausprägungen angegeben werden. Die Definition und Pflege von *Aufzählungsmerkmalen* erfordert zusätzlich das Anlegen der entsprechenden Wertausprägungen.

Merkmalstyp	Klasse
Aufzählungsmerkmal	EnumAT
Freitextmerkmal	TextAT
Datumsmerkmal	DateAT
Verantwortlichkeitsmerkmal	ResponsibilityAT
Zahlmerkmal	NumberAT

Tabelle 2 - Implementierung der Merkmalstypen

Abbildung 8 zeigt das UML-Klassendiagramm der Implementierung des iteraplan Attributmechanismus. Dieser ermöglicht keine Informationsmodellerweiterungen durch das Anlegen neuer Elemente auf Typebene. Vielmehr werden selbst definierte *Merkmale* („*Merkmalstypen“*) durch Instanzen der Klasse *AttributeType*, bzw. einer deren Subklassen repräsentiert. Die Definition eines neuen *Merkmals* wird also durch das Anlegen eines neuen Objekts der Klasse *AttributeType* realisiert. Die Zuordnung des *Merkmals* zu den entsprechenden *Buildingblocktypen* wird über die Beziehung zwischen *AttributeType* und *BuildingBlockType* abgebildet. Tabelle 2 ordnet den *Merkmalstypen* ihren jeweiligen Modellklassen zu.

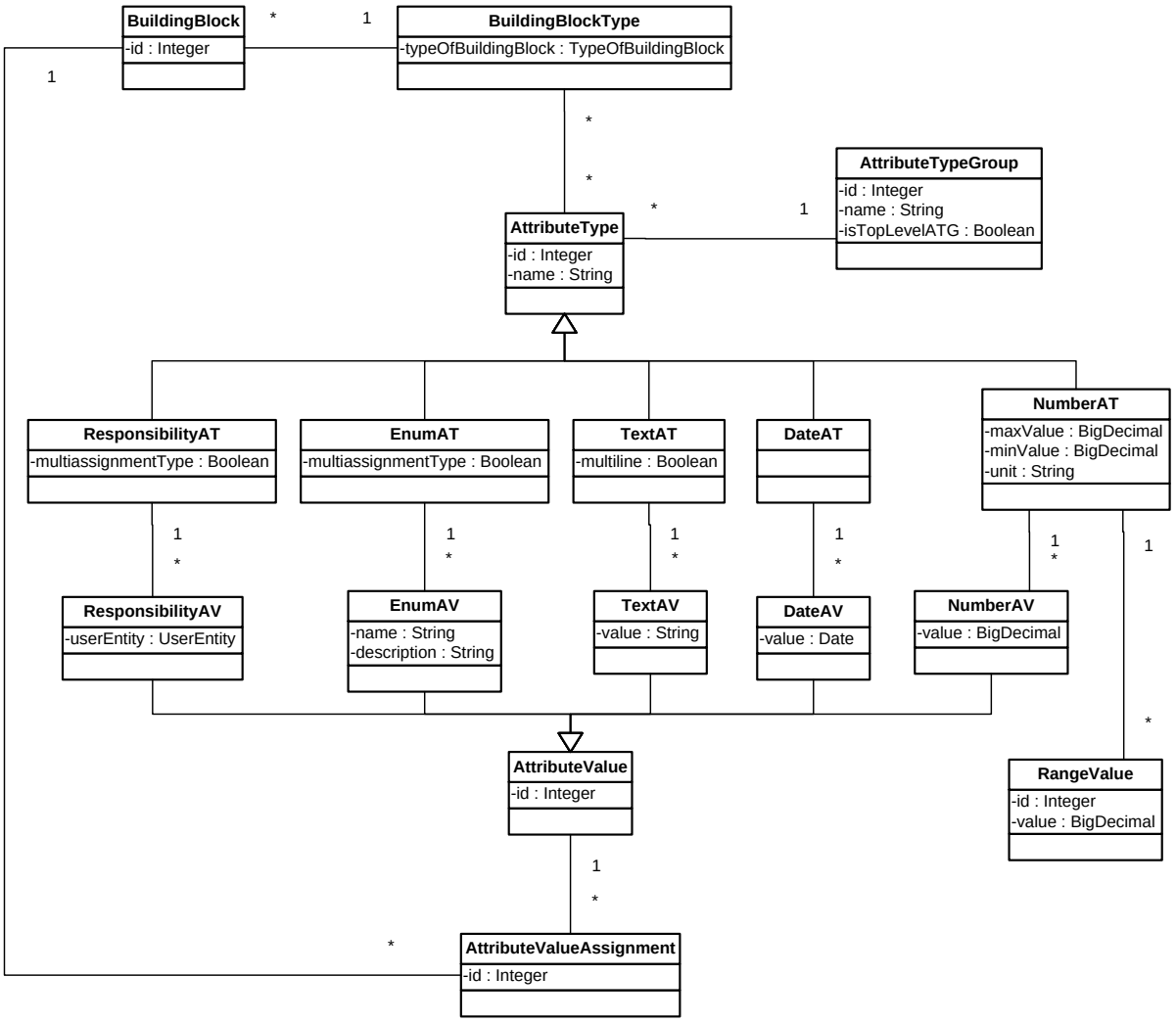

Abbildung 8 - iteraplan Attributmechanismus

Um die Gesamtheit aller *Merkmale* zu strukturieren, können zusätzlich *Merkmalsgruppen* angelegt werden. Diese werden durch die Klasse *AttributeTypeGroup* realisiert. Durch die Zuordnung von *Merkmalen* zu *Merkmalsgruppen* erfolgt so eine Kategorisierung. Durch die Kennzeichnung einer *Merkmalsgruppe* als „Top-Level-Merkmalsgruppe“ (vgl. Attribut „isTopLevelATG“ der Klasse *AttributeTypeGroup*) erfolgt die Darstellung der zugeordneten *Merkmale* in iteraplan auf die gleiche Weise wie Standardattribute des jeweiligen Buildingblocktyps. Somit können derartige *Merkmale* bei der Pflege der zugehörigen *Buildingblocktypen* nicht von Standardattributen unterschieden werden.

Die Pflege der Attributwerte für Instanzen des jeweiligen *Buildingblocktypen* erfolgt über das Anlegen sowohl eines *AttributeValueAssignment*- als auch eines *AttributeValue*-Objekts der entsprechenden Subklasse. Jedem *AttributeValueAssignment*-Objekt ist somit eine *AttributeValue*-Instanz zugeordnet. Die Zuordnung der Werte zum jeweiligen Objekt wird durch die Beziehung zwischen den Klassen *AttributeValueAssignment* und *BuildingBlock* realisiert.

3.4 Anwendung

Die Entwicklung des Werkzeugs als Webanwendung ermöglicht den standortunabhängigen Mehrbenutzerbetrieb. Zur Aufteilung von Verantwortungen sowie zur Gewährleistungen von Sicherheitsaspekten können in iteraplan *Benutzern* oder *Anwendergruppen* Berechtigungen zugeordnet werden, die so Lese- und Schreibrechte auf *Buildingblocktypen*, Auswertungen und andere Funktionen des Werkzeugs definieren.

Um die Unternehmensarchitektur auf iteraplan abzubilden, können Anwender mit entsprechender Berechtigung, Instanzen der verschiedenen *Buildingblocktypen* sowie deren Attribute, Beziehungen und *Merkmale* anlegen und pflegen. Dabei kann über entsprechende Attribute verschiedener *Buildingblocktypen* sowohl der aktuelle Stand, als auch die geplante Unternehmensarchitektur modelliert werden.

Zur Beherrschung der Komplexität der Applikationslandschaft bzw. der gesamten Unternehmensarchitektur, bietet iteraplan eine Reihe von Auswertungs- bzw. Exportmöglichkeiten. Diese ermöglichen es durch einen Abfragemechanismus bestimmte Instanzen verschiedener Buildingblocktypen auszuwählen, diese zu exportieren oder zu visualisieren. Da diese Arbeit jedoch hauptsächlich den o.g. *Attributmechanismus* sowie die allgemeine Adaptionsfähigkeit des Werkzeugs untersucht, wird an dieser Stelle darauf verzichtet, alle Anwendungsmöglichkeiten detailliert vorzustellen. Eine derartige Beschreibung liefert [it10].

4 Erhebung von Anforderungen aus der Praxis

Nachdem der iteraplan *Attributmechanismus* detailliert vorgestellt wurde, wird das Werkzeug nun im Hinblick auf die Anforderungen an die Anpassbarkeit des Informationsmodells untersucht. Um diese Untersuchungen vorzubereiten, werden nachfolgend die möglichen allgemeinen Informationsmodellanpassungen beschrieben sowie im Anschluss daran, auf ihre Unterstützung durch iteraplan geprüft.

4.1 Informationsmodellanpassungen - allgemein

Im Rahmen ihrer Werkzeugstudie untersuchte die TU München verschiedene Werkzeuge für das EAM hinsichtlich funktionaler und nicht-funktionaler Anforderungen (vgl. [Ma08]). Die Studie stützt sich dabei auf sogenannte „Szenarien“, welche konkrete und aus der Praxis gewonnene Anforderungen zusammenfassen. Eines der Szenarien zielt dabei auf die Konfigurier- und Adaptierbarkeit des Informationsmodells im Werkzeug ab. In der Folge werden, ausgehend von diesem Szenario, konkrete Anforderungen abgeleitet und den drei Kategorien *Ändern*, *Löschen* und *Anlegen* zugeordnet (vgl. Abbildung 9).

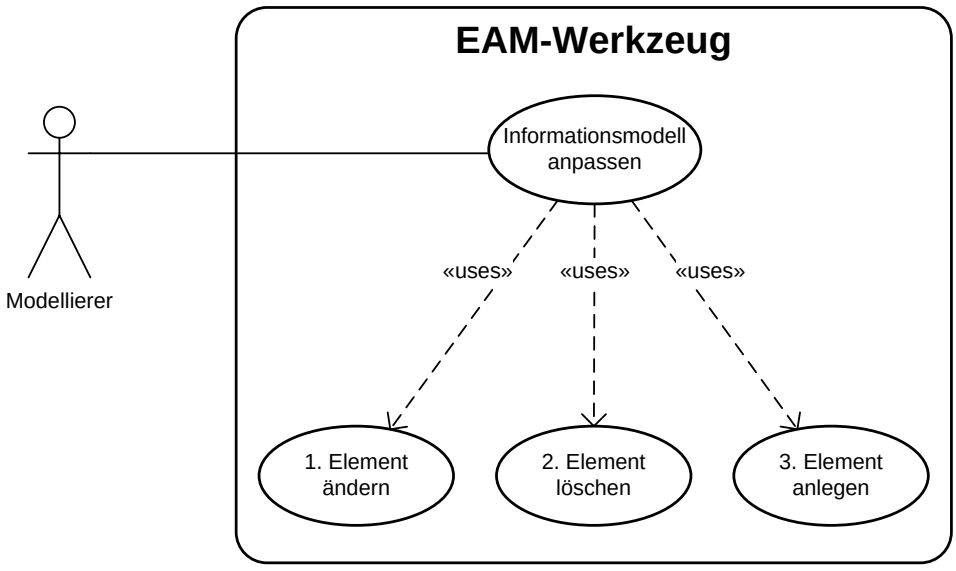

Abbildung 9 – Informationsmodellanpassungen

4.1.1 Ändern

Diese Kategorie umfasst die in Abbildung 10 dargestellten Anwendungsfälle wie das Umbenennen von Klassen, Attributen und Beziehungen eines bestehenden Informationsmodells. Anpassungen dieser Form treten oft auf, wenn sich bereits vor der Einführung des EAM-Werkzeugs eine unternehmensspezifische EA-Terminologie entwickelt hat. Sie dienen dazu, die Akzeptanz des Werkzeugs durch die Benutzer – v.a. während der Einführungszeit – zu

erhöhen. Die Notwendigkeit dieser Umbenennungen ergibt sich aus der bestehenden Vielzahl von Begriffen, welche für die verschiedenen EA-Elementtypen verwendet werden. Ein typisches Beispiel hierfür sind die verbreiteten Begriffe „Informationssystem“ und „Applikation“, die beide den von [Bu08] beschriebenen Elementtyp „BusinessApplication“ bezeichnen. Da sich bis heute keines der in Abschnitt 2 genannten EA-Frameworks durchgesetzt hat, beschränkt sich die Begriffsvielfalt auch nicht nur auf die eben beschriebene Klassenebene. Auch die Beziehungen zwischen einzelnen Klassen können unterschiedlich definiert und benannt werden. Daher sollten EAM-Werkzeuge die Umbenennung von Elementen des Informationsmodells unterstützen. Neben Umbenennungen einzelner Modellelemente existiert noch eine Reihe weiterer Anwendungsfälle, bei denen Elemente des Informationsmodells geändert werden. So können auch Kardinalitäten sowie die Typen von Attributen oder Beziehungen geändert werden. Ein typisches Beispiel hierfür ist die Änderung eines optionalen Attributes, hin zu einem Pflichtattribut, was einer Änderung der unteren Schranke der jeweiligen Kardinalität von 0 zu 1 entspricht.

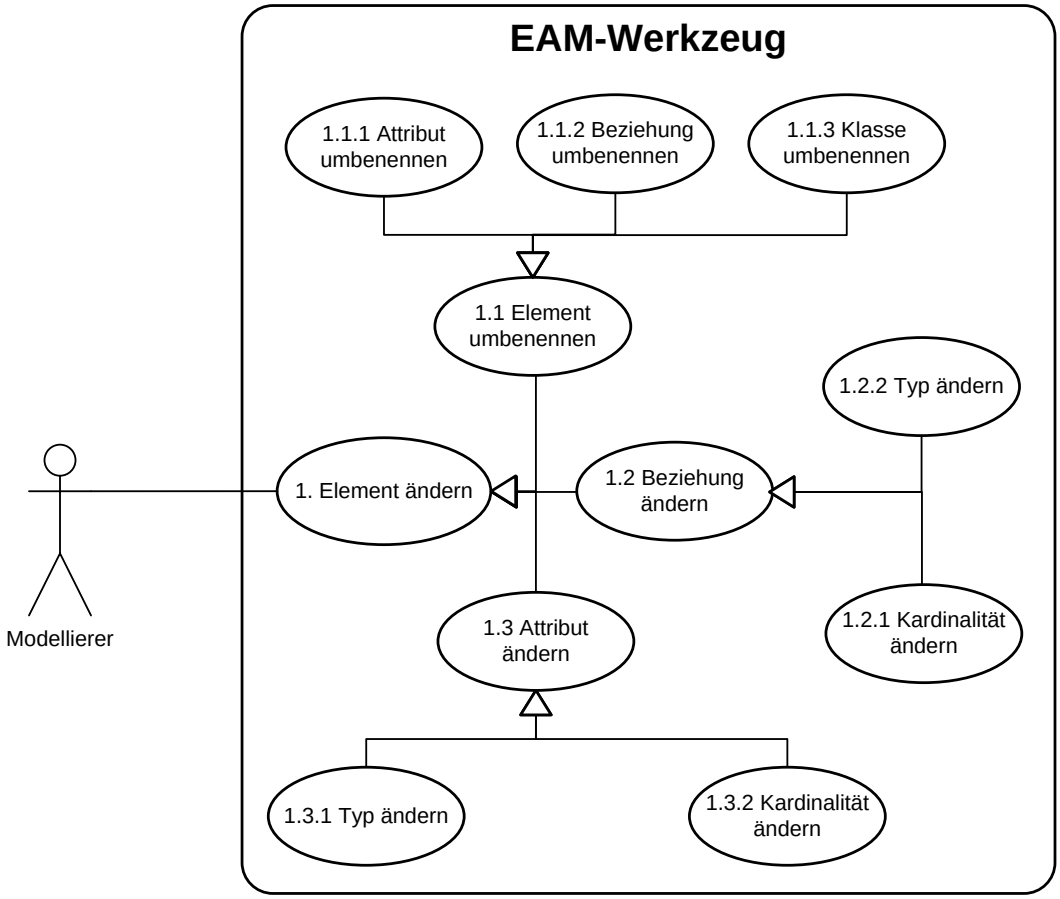

Abbildung 10 – Informationsmodelländerungen

4.1.2 Löschen

Im Gegensatz zu den soeben vorgestellten *Informationsmodelländerungen* beinhaltet die Kategorie *Löschen* Anpassungen, bei denen Elemente des vorliegenden Informationsmodells entfernt werden (vgl. Anwendungsfälle in Abbildung 11). Dies kann vorgenommen werden, wenn das vorliegende Informationsmodell EA-Konzepte berücksichtigt, die vom Unternehmen als irrelevant eingestuft werden. Dabei können Unternehmen sich sowohl gegen die Nutzung ganzer Elementtypen und deren Beziehungen entscheiden, als auch gegen die Pflege bzw. Verwendung einzelner Relationen oder Attribute.

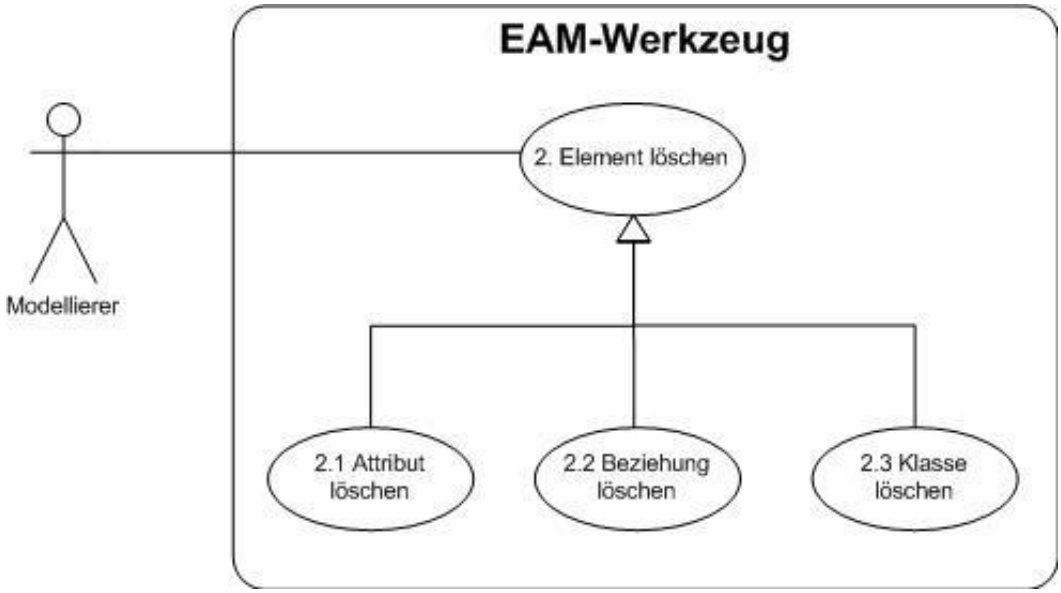

Abbildung 11 – Informationsmodellreduzierungen

Besonderes Interesse bei derartigen Anpassungen gilt dabei dem Verhalten des Werkzeugs, wenn zur Zeit des Löschens bereits Instanzen bzw. Werte für das jeweilige Element angelegt wurden. Denkbar wäre hier das Verbieten des Löschens (vgl. Abbildung 12 – Option A), das Ausblenden des Elements auf der Oberfläche bei gleichzeitiger Beibehaltung der entsprechenden Daten (vgl. Abbildung 12 – Option B), sowie das Entfernen des Elements von der Oberfläche, bei gleichzeitigem Löschen der zugehörigen Daten (vgl. Abbildung 12 – Option C).

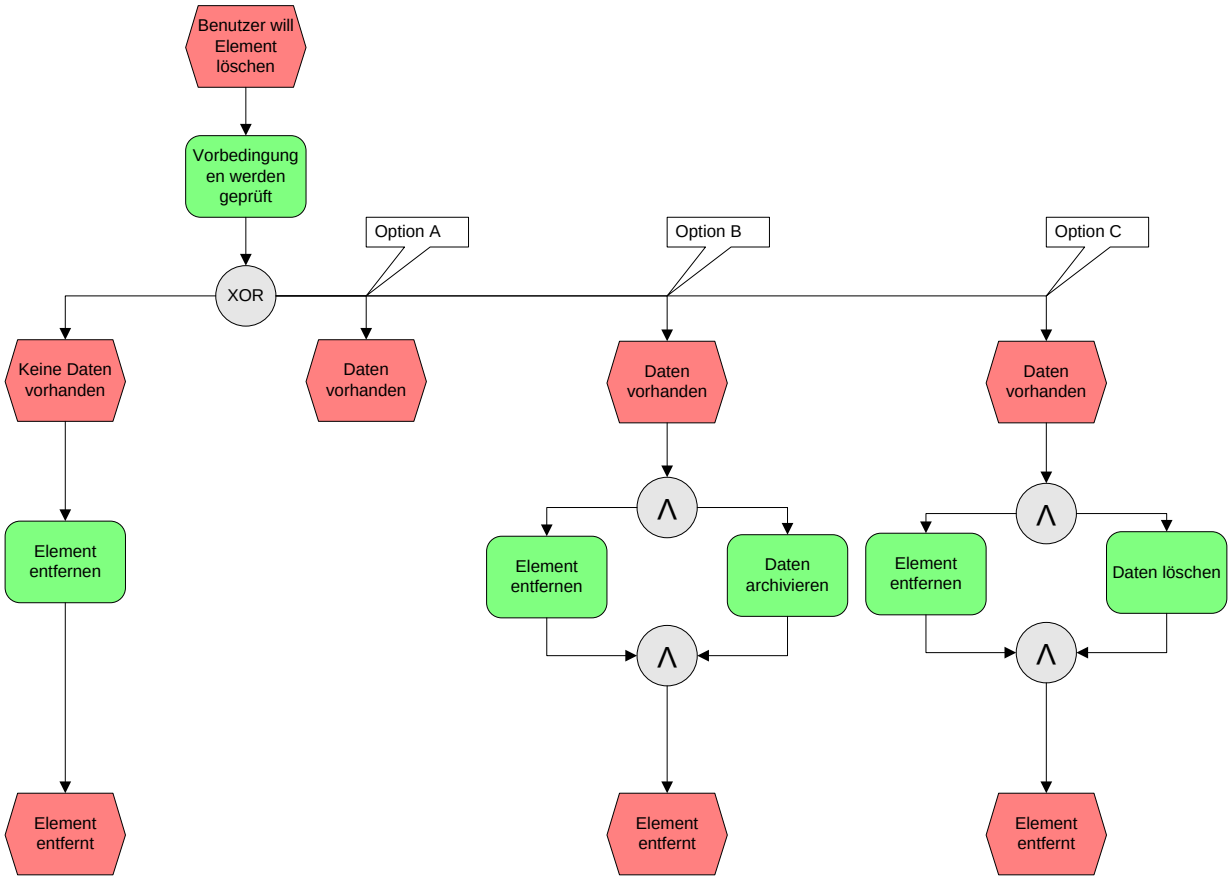

Abbildung 12 – Reduzierung des Informationsmodells – Optionen

4.1.3 Anlegen

Eine besonders wichtige Kategorie von Anforderungen umfasst das *Anlegen* neuer Elemente. Sie beschreibt alle Anwendungsfälle, welche die Definition neuer Attribute, Relationen und Klassen beinhalten (vgl. Abbildung 13). Der einfachste dieser Anwendungsfälle ist das Anlegen eines neuen Attributs in einer bestehenden Klasse. Dies ist bei EAM-Werkzeugen immer dann notwendig, wenn ein Unternehmen beabsichtigt, zusätzliche Informationen zu einem Elementtyp zu pflegen. Zu beachten ist hierbei, inwieweit das Werkzeug dabei die Modellierung von Konsistenzbedingungen sowie deren Überprüfung im Hinblick auf Standardwerte, Pflichtmerkmale und Typ des Attributs unterstützt. Gleiches gilt für die Definition neuer Beziehungen. Wie bereits in Kapitel 2 beschrieben, gibt es verschiedene Auffassungen von EA-Konzepten, Elementtypen und deren Beziehungen untereinander. Somit besteht eine wesentliche Anforderung an die Anpassbarkeit von Informationsmodellen eines EAM-Werkzeugs in der Möglichkeit, neue Beziehungen zwischen den Modellklassen der entsprechenden Elementtypen zu definieren. Auch hier gilt der Modellierung von Konsistenzbedingungen sowie deren Überprüfungen besonderes Interesse. Neben dem Anlegen neuer Attribute und Beziehungen, können auch Fälle auftreten, in denen das Informationsmodell eines EAM-

Werkzeugs um ganze Klassen erweitert werden muss. Diese Möglichkeit, und die damit verbundene Flexibilität bei der Gestaltung des Informationsmodells, werden meist nur von metamodellorientierten Werkzeugten unterstützt.

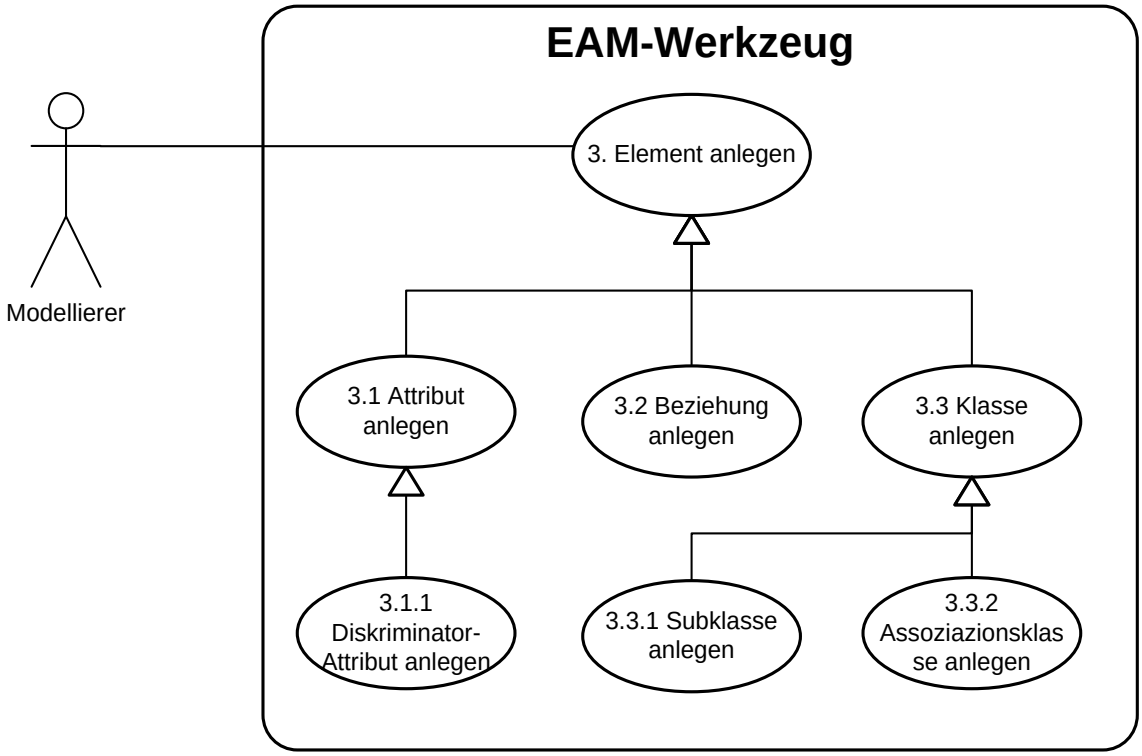

Abbildung 13 – Informationsmodellerweiterungen

Eine besondere Erweiterung im Zuge des Anlegens eines neuen Attributs liegt vor, wenn dies zur Attributierung einer Beziehung geschieht. Hierbei müssen drei verschiedene Fälle bezüglich der Kardinalität der Beziehung differenziert werden: Bei Beziehungen mit der Kardinalität 1:1 gilt, dass die entsprechende Information in der Datenbank durch die Bildung eines Feldes, in einer der beiden an der betrachteten Beziehung beteiligten Klassen, abgebildet wird. Ähnliches gilt für 1:*-Beziehungen. Hierbei wird das verweisende Feld, in der mit Kardinalität 1 an der Beziehung beteiligten Klasse gebildet. Somit besteht in beiden Fällen die Möglichkeit, die Attributierung der Beziehung durch das Anlegen eines weiteren Attributs in einer der beiden entsprechenden Klassen vorzunehmen. Der komplexere Fall ist die Attributierung einer *:*-Beziehung. Da derartige Beziehungen nicht wie zuvor beschrieben aufgelöst werden können, ist hierfür die Bildung einer Assoziationsklasse nötig (vgl. Anwendungsfall 3.3). Für die Attributierung der Beziehung muss zusätzlich ein weiteres Attribut in der entsprechenden Assoziationsklasse definiert werden (vgl. Abbildung 14).

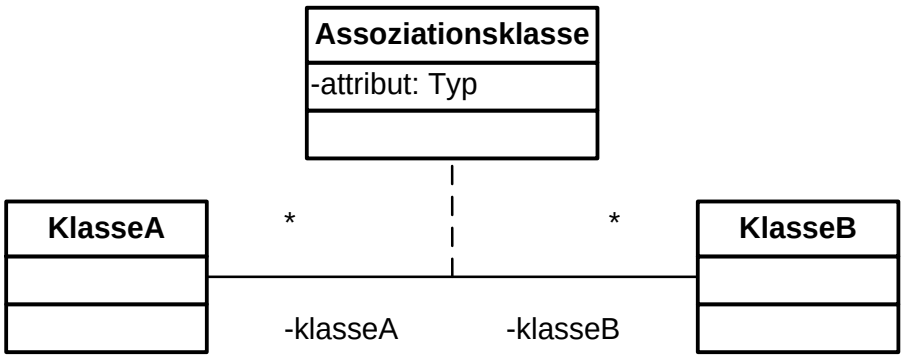

Abbildung 14 - Attributierung von Beziehungen

Ein weiterer denkbarer Spezialfall einer Informationsmodellerweiterung, ist die Typisierung durch Modellierung einer Vererbungsstruktur wie sie in Abbildung 15 dargestellt ist. Dies bedeutet, dass Klassen weiter untergliedert werden, und Attribute sowie Beziehungen speziell für einzelne Subtypen definiert werden können. Als Beispiel hierfür ist die Modellierung von Informationssystemen, bei denen zwischen Standard- bzw. Individualsoftware unterschieden werden kann, zu nennen. Denkbare Attribute für Standardsoftware sind z.B. „Lizenzkosten“, wohingegen bei der Individualsoftware weiter zwischen Eigen- und Fremdentwicklung differenziert werden kann.

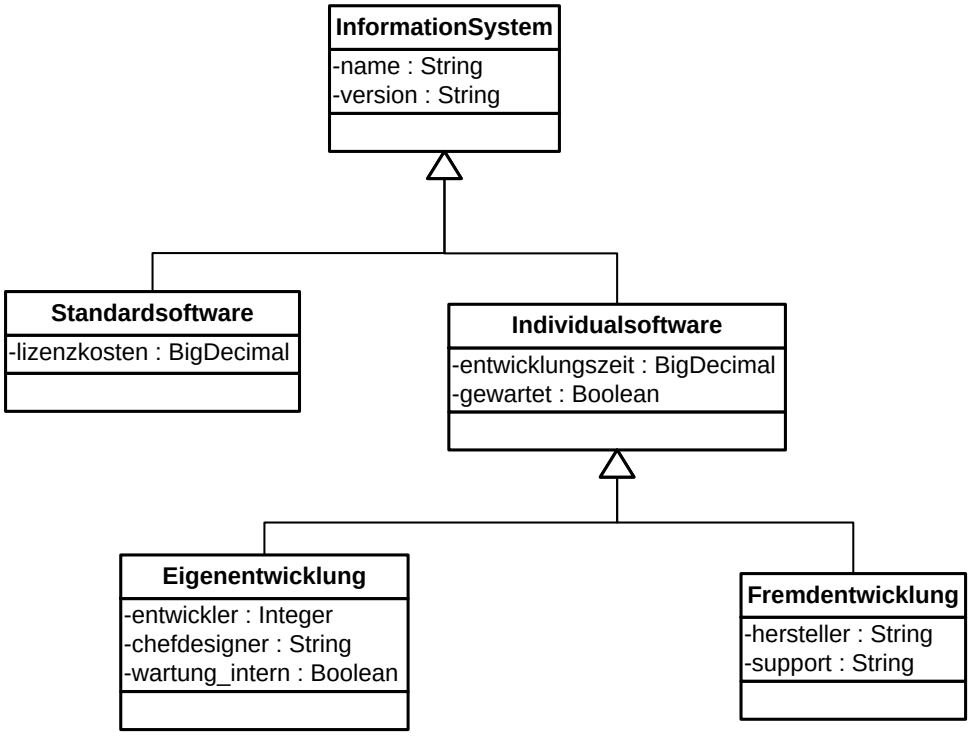

Abbildung 15 - Subklassenbildung

Der letzte hier aufgeführte Spezialfall einer Informationsmodellerweiterung besteht in einer *orthogonalen Typisierung*. Diese liegt vor, wenn Instanzen einer Klasse mehrere Status annehmen können. Die Typisierung lässt sich am Beispiel der Projekte darstellen, die im Status „Vorschlag“ angelegt werden und nacheinander die Staus „genehmigt“, „durchgeführt“ und

„abgeschlossen“ durchlaufen. Eine ähnliche Menge an Status ist auch für Informationssysteme denkbar, die wie in Abbildung 16 dargestellt von „spezifiziert“ über „implementiert“ und „getestet“ zu „aktiv“ übergehen.

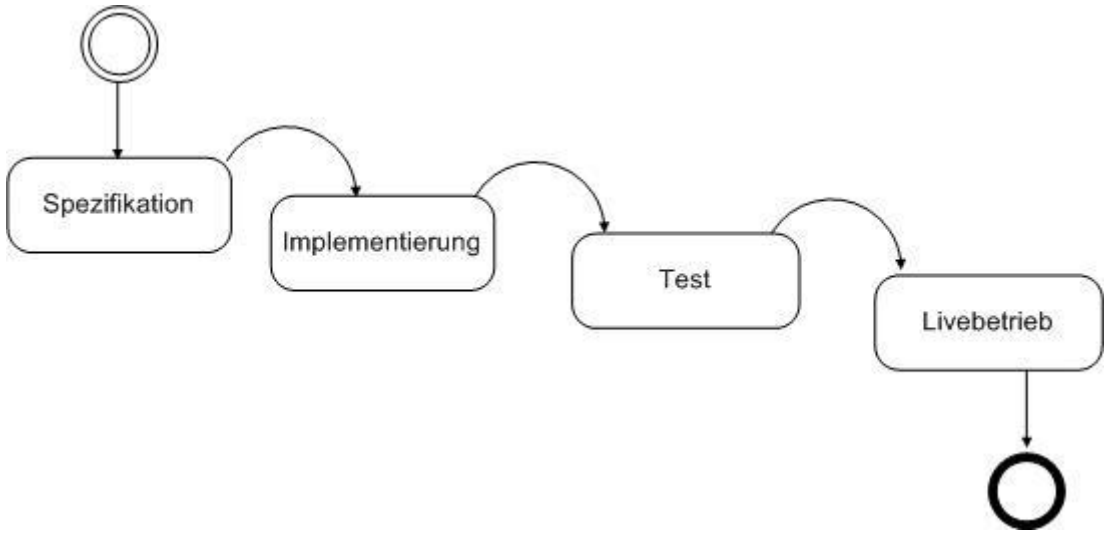

Abbildung 16 – Orthogonale Typisierung

Orthogonale Typisierung lässt sich durch das Anlegen eines Attributs, welches auf den jeweiligen Zustand verweist, oder die Definition von zustandsbasierten Klassen, bei gleichzeitiger Modellierung entsprechender Bedingungen für die Zustandsübergänge realisieren.

4.2 Anpassungsmöglichkeiten in iteraplan

Nachdem die allgemeinen Anforderungen an die Anpassbarkeit von Informationsmodellen von EAM-Werkzeugen beschrieben wurden, kann nun untersucht werden inwieweit diese von iteraplan unterstützt werden. Hierzu werden nachfolgend die im vorigen Abschnitt beschriebenen Informationsmodellanpassungen auf ihre Realisierbarkeit in iteraplan untersucht. Die Ergebnisse der Analyse werden neben der textuellen Beschreibung graphisch, mithilfe annotierter *UseCase*-Diagramme dargestellt. Im Detail werden die einzelnen Anwendungsfälle in den Farben *Grün*, *Orange* und *Rot* eingefärbt. *Grün* bedeutet dabei die uneingeschränkte, *Orange* eine eingeschränkte und *Rot* keine Unterstützung der jeweiligen Anpassung durch das Werkzeug.

4.2.1 Ändern

iteraplan unterstützt das *Ändern* von Elementen des Informationsmodells in eingeschränkter Form: Das *Umbenennen* von Klassen, Attributen und Beziehungen ist möglich, wenngleich hierbei keine Änderungen direkt im Informationsmodell vorgenommen werden. Es besteht jedoch die Möglichkeit, die dargestellten Namen auf der Präsentationsschicht des Werkzeugs, an die unternehmensspezifische Terminologie anzupassen. Diese Änderungen können dabei sowohl bei der Installation, als auch zu späteren Zeitpunkten in Konfigurationsdateien vorgenommen werden. Die Änderung von Beziehungen ist dabei ausschließlich über derartige Umbenennungen möglich. Das Ändern eines Attributs (1.3) ist dagegen eingeschränkt möglich (vgl. Abbildung 17). Die Änderung des Typs eines Attributs (1.3.1) wird nicht unterstützt. Anwendungsfall 1.3.2 wird dagegen für bestimmte Fälle unterstützt. Die Pflege von *Merkmalstypen* des *Attributmechanismus* erlaubt die Änderung der unteren Schranke von 0 zu 1, sowie von 1 zu 0. Damit kann für jeden *Merkmalstyp* sowohl bei der Definition, als auch nachträglich, entschieden werden, ob es sich dabei um ein Pflichtmerkmal handelt oder nicht. Für selbst definierte Merkmale vom Typ *Aufzählungs-* bzw. *Verantwortlichkeitsmerkmal* kann zusätzlich entschieden werden, ob eine mehrfache Wertzuordnung zulässig ist. Die Änderung der oberen Schranke von *1* zu * ist dabei zu jeder Zeit möglich. Die bereits persistierten Daten bleiben dabei erhalten. Das Ändern der oberen Schranke in umgekehrter Weise ist dagegen nur möglich, wenn es zur Zeit der Änderung keine Instanzen gibt, für die mehr als ein Wert eingepflegt wurde. Auch hier bleiben nach erfolgter Änderung alle bereits zuvor gepflegten Daten erhalten.

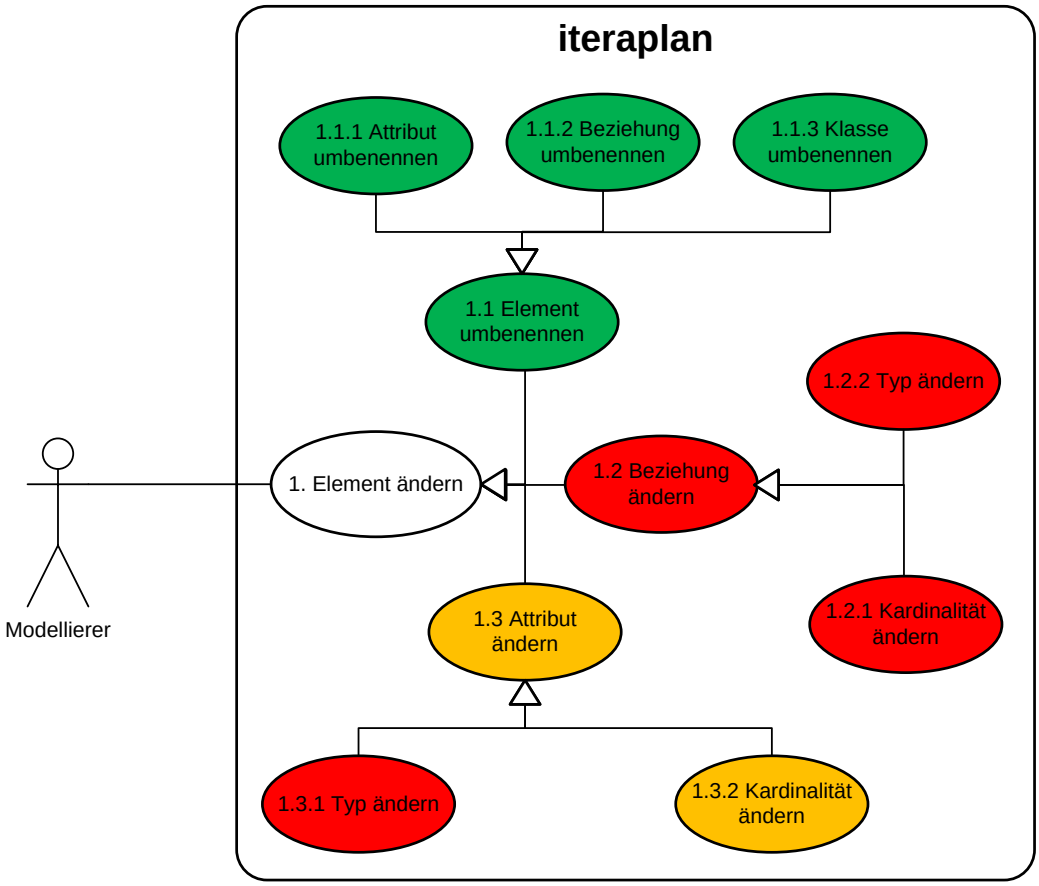

Abbildung 17 – Informationsmodelländerungen – iteraplan

4.2.2 Löschen

iteraplan unterstützt alle aufgeführten Anwendungsfälle einer Informationsmodellreduzierung (vgl. Abbildung 18). Jedoch stellen auch diese Reduzierungen keine Änderung am eigentlichen Informationsmodell dar. Informationsmodellreduzierungen können in iteraplan über das Berechtigungskonzept des Werkzeugs realisiert werden. Die Rechteverwaltung erlaubt Administratoren u.a. Lese- bzw. Sichtbarkeitsrechte auf *Buildingblock*- bzw. Attribut- und Beziehungsebene zu definieren. Eine unternehmensweite Entscheidung gegen die Nutzung bestimmter Elemente des iteraplan-Informationsmodells, kann also über eine Ausblendung dieser Elemente für alle Benutzer erfolgen. Da die Reduzierung des Informationsmodells um ein von iteraplan definiertes Element lediglich zur Ausblendung führt, bleiben hierbei bereits persistierte Daten erhalten. Abbildung 18 indiziert durch eine Orangefärbung eine Einschränkung des Anwendungsfalls 2.2. Diese liegt darin, dass im Werkzeug lediglich Beziehungen zwischen verschiedenen Beziehungen auf die beschriebene Weise aus dem Informationsmodell entfernt werden können. Für Autoassoziationen trifft dies jedoch nicht zu.

Dagegen können *Merkmalstypen*, die über den in Abschnitt 3.3 beschriebenen *Attributmechanismus* angelegt wurden, auch wieder gelöscht werden. Hierbei gehen, im Gegensatz zur zu-

vor behandelten Ausblendung einzelner Informationsmodellelemente über das Berechtigungskonzept, alle bereits persistierten Daten verloren.

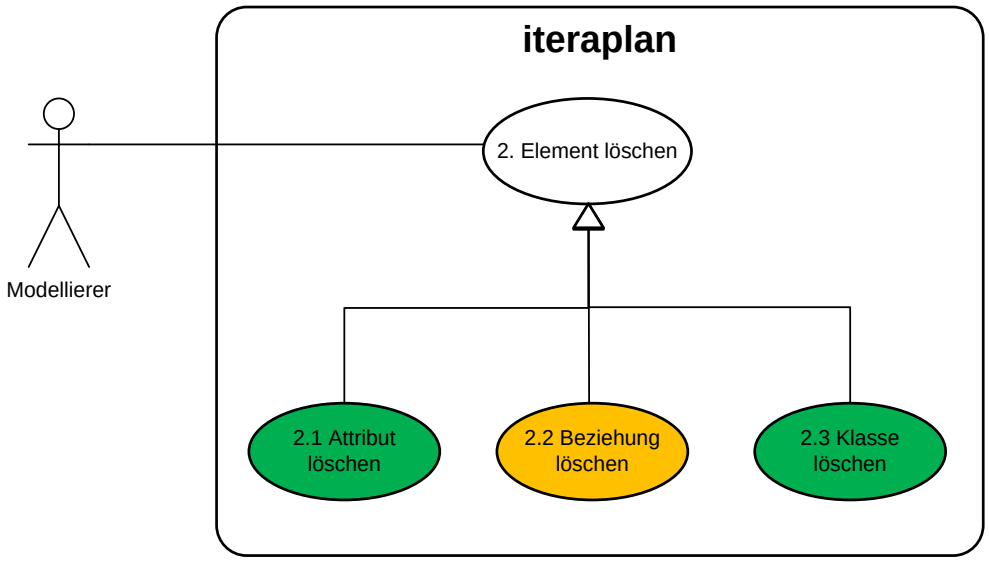

Abbildung 18 – Informationsmodellreduzierungen – iteraplan

4.2.3 Anlegen

Die Definition neuer Elemente des Informationsmodells wird von iteraplan (Version 2.6) durch die Definition zusätzlicher Attribute über den in Abschnitt 3.3 beschriebenen *Attributmechanismus* unterstützt. Eine Informationsmodellerweiterung durch das Hinzufügen neuer Klassen und Beziehungen wird dagegen nicht unterstützt.

Auch für die in Abschnitt 4.1.3 beschriebenen Spezialfälle einer Informationsmodellerweiterung, gelten in iteraplan gewisse Einschränkungen: Die Attributierung einer Beziehung kann nur dann vorgenommen werden, wenn es sich um eine Beziehung der Kardinalität 1:1 bzw. 1:* handelt. Hierzu wird über den *Attributmechanismus* ein zusätzliches *Merkmal* definiert, das wie in Abschnitt 4.1.3 beschrieben, die gewünschte Information abbildet. Für die Attributierung von *:*-Beziehungen können in iteraplan zusätzliche *Merkmale* für den *Buildingblocktyp Fachliche Zuordnung* definiert werden. Somit können die Beziehungen zwischen *Geschäftsprozessen*, *Geschäftseinheiten*, *Produkten* und *Informationssystemen* über den *Attributmechanismus* erweitert werden. Die verbleibenden *:*-Beziehungen des iteraplan-Informationsmodells (vgl. Abbildung 7) können dagegen nicht attributiert werden.

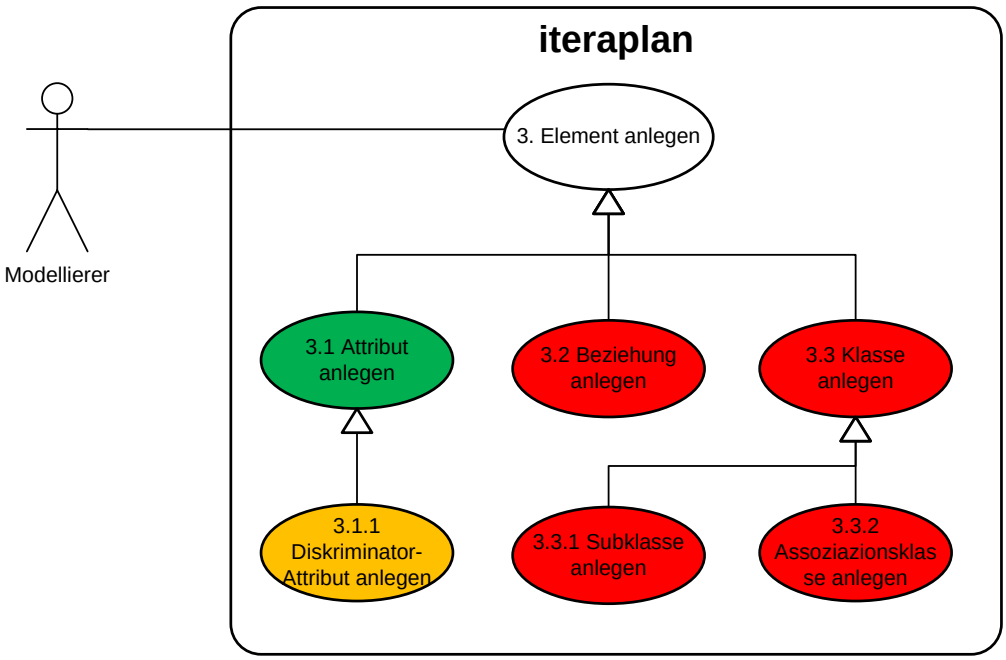

Abbildung 19 – Informationsmodellerweiterungen – iteraplan

Da von iteraplan die Definition neuer Klassen im Informationsmodell nicht unterstütz wird, kann eine Typisierung nur durch die Benutzung sog. „Diskriminatorattribute“ oder entsprechender Modellierungsvereinbarungen erreicht werden. Denkbare Lösungen hierfür bestehen aus Namenskonventionen sowie *Aufzählungsmerkmalen* des *Attributmechanismus*, die auf den jeweiligen Subtyp des *Buildingblocktyps* verweisen. Gleiches gilt auch für die in Abschnitt 4.1.3 beschriebene *Orthogonale Typisierung*. Diese kann in iteraplan durch die Bildung eines *Aufzählungsmerkmals* mit Wertausprägungen, die den möglichen Zuständen entsprechen, abgebildet werden.

4.3 Analyse vordefinierter EA-Informationsmodelle

Nachdem nun sowohl die allgemeinen Anforderungen an die Informationsmodellanpassungen von EAM-Werkzeugen, als auch die von iteraplan unterstützten Informationsmodellanpassungsmöglichkeiten bekannt sind, werden in diesem Abschnitt Informationsmodelle verschiedener Unternehmen – im Folgenden als Organisation#X bezeichnet – betrachtet. Diese Anwender haben bereits vor der Einführung von iteraplan ein eigenes Informationsmodell der zu modellierenden Unternehmensarchitektur erstellt. Diese Modelle werden nachfolgend einzeln auf ihre Übertragbarkeit auf das iteraplan Metamodell bzw. das iteratec EA-Framework hin untersucht. Besonderes Interesse gilt dabei der Identifikation signifikanter Differenzen zwischen den Modellen die dazu führen, dass das Informationsmodell von iteraplan an die unternehmensspezifischen Aspekte des Architekturmodells bei einer Nutzung des Werkzeugs angepasst werden muss.

4.3.1 Organisation#1

Das erste untersuchte Informationsmodell einer Unternehmensarchitektur, wurde im Rahmen einer EAM-Werkzeugauswahl von einem Finanzdienstleistungsunternehmen – im Folgenden als Organisation#1 bezeichnet – entwickelt. Bevor detailliert auf das Informationsmodell und das dazugehörige Anforderungsdokument dieses Unternehmens eingegangen wird, werden die bei der Entwicklung des Modells definierten Architekturebenen näher betrachtet.

Organisation#1 untergliedert seine EA in fünf Architekturebenen: *Applikations-*, *Daten-*, *Geschäftsprozess-*, *Service-* sowie *Technologiearchitektur*. Diese Architekturebenen lassen sich teilweise auf die Ebenen des EA-Frameworks von iteraplan abbilden (vgl. Tabelle 3). Einige der im Informationsmodell enthaltenen Klassen (vgl. Abbildung 20) wurden von Organisation#1 in einem beschreibenden Anforderungsdokument Architekturebenen zugeordnet (vgl. Tabelle 4). Bevor das Informationsmodell detailliert beschrieben wird, sollen in der Folge die auftretenden Abweichungen hinsichtlich der Ebenenstruktur diskutiert werden.

Organisation#1	**iteraplan**
Geschäftsprozessarchitektur	Geschäftsarchitektur
Datenarchitektur	Informationssystem-/Geschäfts-Architektur
Applikationsarchitektur, Servicearchitektur	Informationssystem-Architektur
Technologiearchitektur	Betriebsinfrastruktur-Architektur
	Technische Architektur
	IT-Strategie
	Unternehmensstrategie

Tabelle 3 – Architekturebenenzuordnung Organisation#1

Das EA-Modell von Organisation#1 beinhaltet im Gegensatz zu dem von der iteratec GmbH entwickelten, keine Architekturebene, auf die eine *Unternehmens*- bzw. *IT-Strategie* abgebildet werden kann. Außerdem beinhaltet es keine Ebene, die der *Technischen Architektur* des iteratec-Frameworks entspricht. Die Architekturebenen *Geschäftsprozessarchitektur* und *Technologiearchitektur* finden sich in beiden Ebenenmodellen wieder, da sie mit den Ebenen der *Geschäftsarchitektur* bzw. der *Betriebsinfrastruktur-Architektur* ihre Entsprechung im iteratec EA-Framework haben. Die von Organisation#1 definierte *Datenarchitektur* hat dagegen keine direkt korrespondierende Architekturebene im Framework. Da diese jedoch u.a. die Klassen *GDT* („Geschäftsdatentyp"), *Outputmessage*, sowie *Inputmessage* beinhalten, und somit die Informationen über die von Applikationen übertragenen Daten abbilden, entspricht die *Datenarchitektur* in Teilen der *Informationssystemarchitektur* sowie der *Geschäftsarchitektur* mit ihren Elementen *Informations*- bzw. *Geschäftsobjekt*.

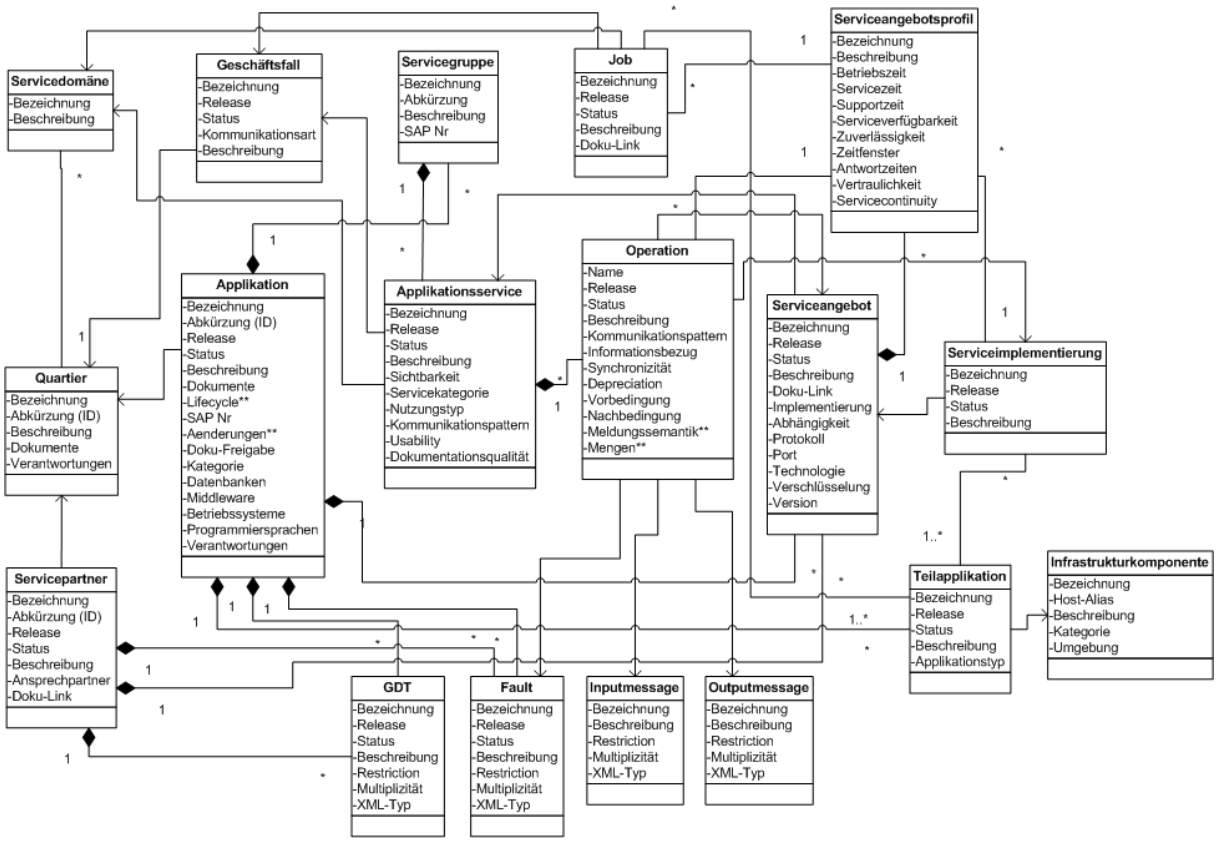

Abbildung 20 – EA-Informationsmodell Organisation#1
(Quelle: iteratec GmbH)

Ein letzter Unterschied in Bezug auf die Architekturebenen liegt in der von Organisation#1 eingeführten Ebenen *Applikationsarchitektur* sowie der *Servicearchitektur*. Diese beinhalten u.a. die Klassen *Applikation* bzw. *Service* und bilden daher die SOA der Organisation#1 ab. Daher werden auch diese beiden Architekturebenen der *Informationssystem-Architektur* des iteratec-EAM-Frameworks zugeordnet. (Das SOA-Konzept, sowie dessen entsprechende EAM-Modellierung wird in Kapitel 5.2.1 detailliert behandelt.)

Das EA-Informationsmodell von Organisation#1 ist in Abbildung 20 als UML-Klassendiagramm dargestellt. Es zeigt die erste Version des Informationsmodells, welches ausschließlich durch die IT-Abteilung der Organisation entwickelt wurde.

Organisation#1 - Architekturebenen und deren Klassen	
Applikationsarchitektur: • *Applikation* • *Quartier* • *Servicepartner* • *Teilapplikation*	Servicearchitektur: • *Applikationsservice* • *Job* • *Operation* • *Serviceangebot* • *Servicedomäne*
Datenarchitektur: • *Geschäftsdatentyp*	Technologiearchitektur: • *Infrastrukturkomponente*
Geschäftsprozessarchitektur: • *Geschäftsfall* • *Teilprozess* • *Teilprozessschritt*	

Tabelle 4 – Architekturebenen Organisation#1

Tabelle 4 ordnet die Klassen des EA-Informationsmodells den bereits beschriebenen Architekturebenen zu, wohingegen in Tabelle 5 die von Organisation#1 definierten Klassen ihrer Entsprechung im Informationsmodell von iteraplan zugeordnet werden. Kombiniert man nun die Erkenntnisse aus diesen Tabellen ergeben sich weitere Unterschiede zwischen den betrachteten Informationsmodellen.

Ein erster Unterschied besteht in der kundenspezifischen Terminologie. So werden beispielsweise *Informationssysteme* als *Applikationen* bezeichnet, *Geschäftsobjekte* als *Geschäftsdatentypen*, *Architekturdomänen* als *Servicedomänen* usw. Ein weiterer Unterschied, zeigt sich durch die mehrfache Abbildung von Klassen der Organisation#1 auf den iteraplan-*Buildingblocktyp Informationssystem*. Dies liegt u.a. an der bereits benannten Modellierung einer *Serviceorientierten Architektur* durch Organisation#1, sowie der Modellierung dieses Konzepts in iteraplan (vgl. Abschnitt 5.2.1). Das Werkzeug stellt keine *Buildingblocktypen* bereit, die in direkter Weise den Klassen *Serviceangebot*, *Serviceangebotsprofil*, *Serviceimplementierung* und *Servicegruppe* entsprechen. Diese können jedoch *Informationssystemen* zugeordnet werden. Über passende Erweiterungen, kann die gewünschte Information durch geeignete *Merkmale* auf diesen *Buildingblocktyp* abgebildet werden. So kann beispielsweise die durch die Klasse *Serviceangebotsprofil* abgebildete Information über ein *Aufzählungsmerkmal* „SLA“ (Service-Level-Agreement) modelliert werden. Dies zeigt, genau wie der weitere Vergleich der Klas-

sen bzw. Architekturebenen, dass Organisation#1 Elementtypen an verschiedenen Stellen feiner differenziert als der Ansatz des Werkzeugs. So werden nach dem iteraplan-Informationsmodell *Geschäftsobjekte* (über *Schnittstellen*) von *Informationssystemen* übertragen, wohingegen nach Verständnis von Organisation#1, verschiedene Informationsobjekte von *Applikationen* übertragen bzw. bei der Ausführung von *Operationen* verarbeitet werden. Neben den *Geschäftsdatentypen* (GDT) die den *Geschäftsobjekten* in iteraplan entsprechen, betrifft dies Objekte der Klassen *Fault*, *Inputmessage* und *Outputmessage*, die alle nach der Zuordnung in Tabelle 5, auf den iteraplan *Buildingblocktyp Geschäftsobjekt* abgebildet werden. Somit besteht ein EAM-Ziel von Organisation#1 in der Beherrschung der Kommunikation bzw. dem Austausch von Informationsobjekten zwischen *Applikationen* und *Operationen.* Dies wird auch durch die eigens hierfür definierte Architekturebene der *Datenarchitektur* bestätigt.

Organisation#1	**iteraplan**
Applikation	Informationssystem
Applikationsservice	Informationssystem
Fault	Geschäftsobjekt
Geschäftsdatentyp	Geschäftsobjekt
Geschäftsfall	Geschäftsprozess
Infrastrukturkomponente	Infrastrukturelement
Inputmessage	Geschäftsobjekt
Job	Geschäftsprozess
Operation	Geschäftsprozess
Outputmessage	Geschäftsobjekt
Quartier	Informationssystemdomäne
Serviceangebot	Informationssystem
Serviceangebotsprofil	Informationssystem
Servicedomäne	Architekturdomäne
Servicegruppe	Informationssystem
Serviceimplementierung	Informationssystem
Servicepartner	Informationssystem
Teilapplikation	Informationssystem

Tabelle 5 – Buildingblocktypzuordnung Organisation#1

Durch die Modellierung einer SOA, bestehend aus *Applikationen* und *Applikationsservices*, ergibt sich zusätzlich ein erhöhter Differenzierungsbedarf für den *Buildingblocktyp* des *Informationssystems*. Sowohl *Applikationen* und *Teilapplikationen* als auch *Applikationsservices* werden auf *Informationssysteme* abgebildet. Auch dieser Unterschied kann bereits durch die Differenzen der jeweiligen Architekturebenen erkannt werden. Beim Vergleich der *Applikations-* und *Servicearchitektur* sowie deren Elemente mit der entsprechenden iteratec *Informationssystemarchitektur* (vgl. Tabelle 3, Tabelle 4) ist festzustellen, dass die *Applikationsarchitektur* in gro-

ßen Teilen der *Informationssystemarchitektur* entspricht. Dies trifft jedoch nicht in gleichem Ausmaß auf die von Organisation#1 definierte *Servicearchitektur* zu. Diese wird zwar in Tabelle 3 ebenfalls der *Informationssystemarchitektur* zugeordnet, allerdings bedeutet die Übertragung der Klassen der *Servicearchitektur* auf die entsprechenden Elemente des iteraplan-Informationsmodells eine gewisse Abstraktion.

Zusammenfassend bleibt festzuhalten, dass das von Organisation#1 beschriebene EA- Informationsmodell einen höheren Detaillierungsgrad bezüglich der übertragenen Informationsobjekte (*Datenarchitektur*) als das Informationsmodell von iteraplan aufweist. Außerdem zeigt es den Bedarf der Modellierung von SOAs auf, welche zum Zeitpunkt der Untersuchung von iteraplan nicht direkt unterstützt wird.

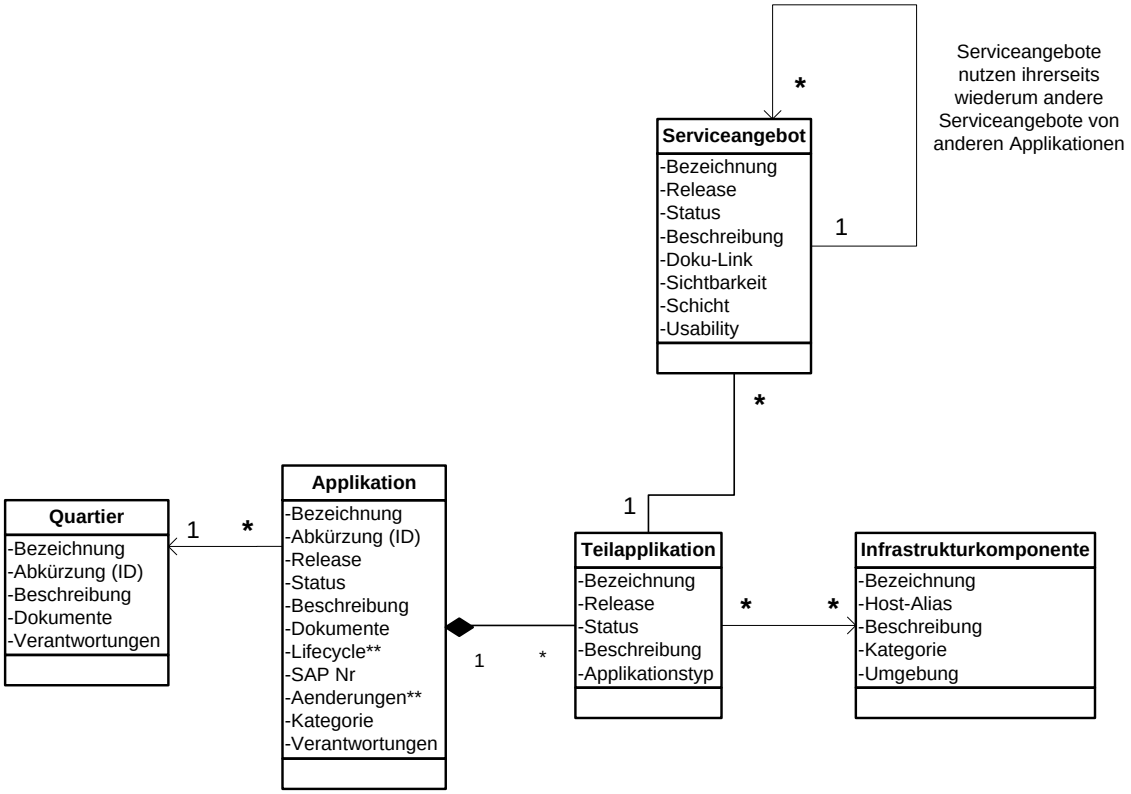

Abbildung 21 – EA-Informationsmodell Organisation#1 – vereinfacht (Quelle: iteratec GmbH)

Im Zuge einer Fokussierung, wurde das in Abbildung 20 vorgestellte Informationsmodell überarbeitet. Im Speziellen wurden hierbei die Ziele identifiziert, welche die Organisation#1 mit Hilfe des EAMs erreichen wollte, um in einem nächsten Schritt relevante und weniger relevante Elemente des bereits entwickelten Informationsmodells differenzieren zu können. Auf Basis dieser Analyse entstand so ein vereinfachtes Informationsmodell (vgl. Abbildung 21), welches im Anschluss als Ausganspunkt für den weiteren Verlauf der EAM-Werkzeugauswahl dienen konnte.

Die Elemente der *Daten-* sowie der *Geschäftsprozessarchitektur* wurden im Zuge der Beratung als irrelevant eingestuft, da diese einen für die EAM-Ziele der Organisation#1 unnötig hohen Detaillierungsgrad aufweisen. Daher wurden diese Klassen und deren Beziehungen nicht in das vereinfachte Informationsmodell übernommen. Die relevanten und daher in das vereinfachte Informationsmodell übernommenen Klassen (*Applikation*, *Infrastrukturkomponente*, *Serviceangebot*, *Teilapplikation* und *Quartier*), besitzen nach Tabelle 5 jeweils eine Entsprechung durch einen von iteraplan bereitgestellten *Buildingblocktyp*. Lediglich die Zuordnung der Klasse *Serviceangebot* zu *Informationssystemen* stellt dabei keine direkte Entsprechung dar. Dies deutet daher u.U. auf einen Bedarf zusätzlicher *Buildingblocktypen* für die Modellierung einer SOA (vgl. Abschnitt 5.2.1) hin.

4.3.2 Organisation#2

Das folgende Modell wurde im Zuge der Vorbereitung einer Unternehmensarchitekturmanagementfunktion von einem Deutschen Versicherungsdienstleisters – nachfolgend als Organisation#2 bezeichnet – entwickelt. Es ist Teil einer Reihe von Anforderungsdokumenten, die zur Evaluierung und Auswahl eines EAM-Werkzeugs herangezogen wurden.

Organisation#2	**iteraplan**
Fachliche Bebauung (Geschäftsarchitektur)	Geschäftsarchitektur
Logische Bebauung (Applikationsarchitektur)	Informationssystem-Architektur
Implementierung (Infrastruktur-Architektur)	Betriebsinfrastruktur-Architektur
Technische Bebauung (Informations-Architektur)	Technische Architektur
Strategie	IT-Strategie
	Unternehmensstrategie

Tabelle 6 – Architekturebenenzuordnung Organisation#2

Das entstandene Informationsmodell ist in Abbildung 22 dargestellt. Im Gegensatz zum bereits beschriebenen Modell der Organisation#1, zeigt der Vergleich des Modells auf Architekturebenen-Ebene nur geringfügige Differenzen auf. Wie das in Abschnitt 3.1 beschriebene EAM-Framework, unterteilt sich das Modell in verschiedene Architekturebenen (vgl. Abbildung 22). Sowohl auf Architekturebenen- als auch auf Klassenebene, bestehen die Unterschiede hauptsächlich aus abweichenden Bezeichnungen der jeweils entsprechenden Elemente. So bezeichnet Organisation#2 beispielsweise *Informationssysteme* als „Applikationen“ und *Informationssystemdomänen* als „Domänencluster“. Zur Visualisierung des Vergleichs werden in Tabelle 6 den von Organisation#2 definierten Architekturebenen, ihre Entsprechungen des EA-Frameworks von iteratec zugeordnet. Daran angelehnt, werden in Tabelle 7 den Elementtypen von Organisation#2 die entsprechenden *Buildingblocktypen* zugeordnet.

Die durch den Vergleich identifizierten Differenzen, liegen in einer abweichenden Zuordnung der Elementtypen zu deren entsprechenden Architekturebenen. So gehört die Klasse *BusinessObject* im iteratec-Framework sowohl zur *Geschäftsarchitektur* als auch zur *Informationssystemarchitektur*, wohingegen gemäß dem Informationsmodell von Organisation#2 *Geschäftsobjekt* ausschließlich der *Fachlichen Bebauung* zugeordnet wird. Noch deutlicher zeigt sich der Unterschied bei *Schnittstellen*. Diese werden von Organisation#2 der *Technischen Bebauung*, und nicht wie die *InformationSystemInterfaces* von iteraplan, der *Informationssystemarchitektur* zugeordnet.

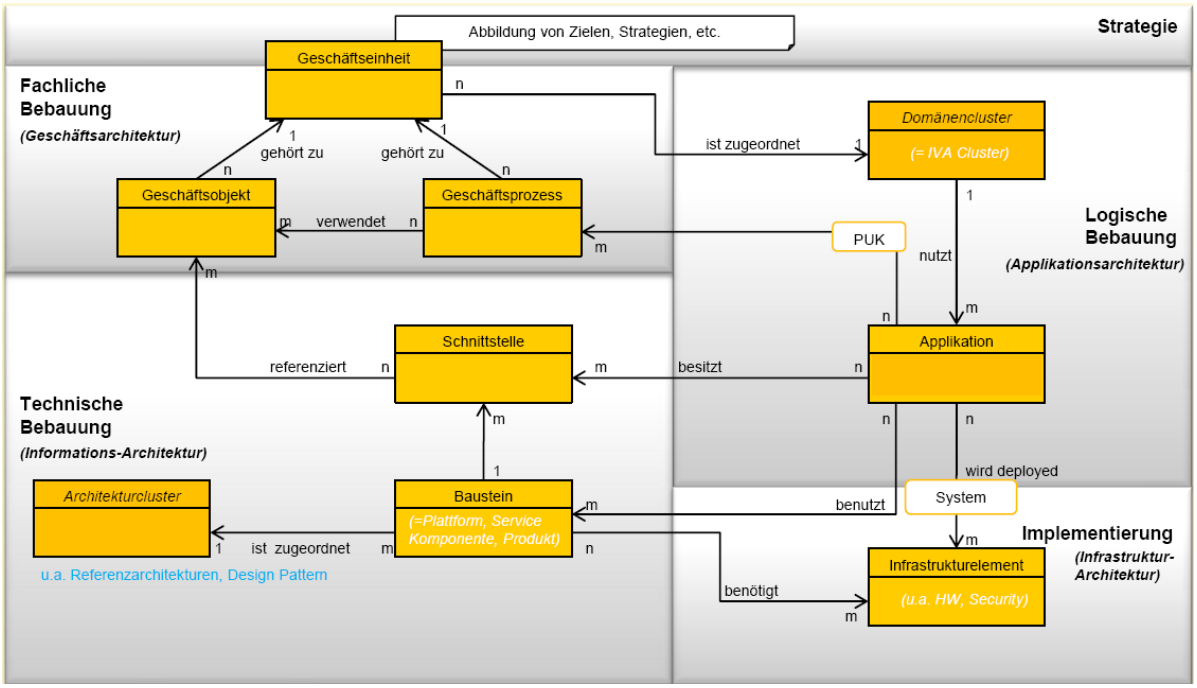

Abbildung 22 – EA-Informationsmodell Organisation#2
(Quelle: iteratec GmbH)

Die Zuordnung von *Applikationen* zu den von ihnen unterstützten *Geschäftsprozessen* geschieht unter Verwendung bereits vorliegender *„Prozessunterstützungskarten“*, was Abbildung 22 durch die Annotation „PUK“ der Beziehung der entsprechenden Elementtypen indiziert.

Ansonsten bleibt lediglich festzuhalten, dass das in Abbildung 22 dargestellte Informationsmodell ausschließlich Elemente enthält, die eine direkte Entsprechung eines *Buildingblocktyps* bzw. einer Beziehung zwischen diesen *Buildingblocktypen* besitzen.

Da das von Organisation#2 definierte Unternehmensarchitekturmodell größtenteils dem iteratec-Framework entspricht, liegt der Schluss nahe, dass das Modell unter Einfluss des von [Ha10] beschriebenen EA-Frameworks entstand.

Organisation#2	iteraplan
Applikation	Informationssystem
Architekturcluster	Architekturdomäne
Baustein	Technischer Baustein
Domänencluster	Informationssystemdomäne
Geschäftseinheit	Geschäftseinheit
Geschäftsobjekt	Geschäftsobjekt
Geschäftsprozess	Geschäftsprozess
Infrastrukturelement	Infrastrukturelement
Schnittstelle	Schnittstelle

Tabelle 7 – Buildingblocktypzuordnung Organisation#2

4.3.3 Organisation#3

Da das nachfolgend untersuchte EA-Informationsmodell der iteratec GmbH ohne beschreibende Dokumente zur Verfügung gestellt wurde, beschränkt sich der Vergleich des Informationsmodells dieses Unternehmens („Organisation#3“) auf Klassenebene.

Organisation#3	iteraplan
Application	Informationssystem
Application Domain	Informationssystemdomäne
Business Area	Geschäftsprozess
Business Component	Informationssystem
Interface	Schnittstelle
Interface Type	Schnittstelle
IT System	Informationssystem
IT System Class	Informationssystem
IT System Track	Informationssystem
IT-Product	Produkt
Organizational Unit	Geschäftseinheit
Person	
Role Type	
Technical Component	Technischer Baustein

Tabelle 8 – Buildingblocktypzuordnung Organisation#3

Tabelle 8 ordnet die in Abbildung 23 eingeführten EA-Elementtypen ihren entsprechenden iteraplan-*Buildingblocktypen* zu. Dieser Vergleich zeigt lediglich einen erhöhten Detaillierungsgrad für *Informationssysteme* und *Schnittstellen*. Der Elementtyp *Interface* entspricht dabei direkt dem iteraplan-*Buildingblocktyp* der *Schnittstelle*, wohingegen *Interface Type* in iteraplan durch einen entsprechenden *Merkmalstyp* abgebildet werden kann. Ähnlich entspricht der Elementtyp *Application* dem *Buildingblocktyp Informationssystem*. Die Elementtypen *IT System Class* sowie *IT System Track* sind dabei ebenfalls über *Merkmalstypen* abbildbar.

Die Elementtypen *Business Component* und *IT System* dienen zur zusätzlichen Unterscheidung zwischen fachlichen und logischen *Informationssystemen*, die jedoch nach [Ha10] auf den *Buildingblocktyp Informationssystem* abgebildet werden können. Für die von Organisation#3 definierten Elementtypen *Person* bzw. *Role Type* existieren keine entsprechenden *Buildingblocktypen*. Die darin enthaltenen Informationen werden in iteraplan über die „Benutzer-, Rollen- und Rechteverwaltung“ abgebildet (vgl. [it10]). Hierzu können Elemente vom Typ „Anwender“ oder „Anwendergruppen“ angelegt und gepflegt werden. Die Definition von Rollen (vgl. *Role Type*) erfolgt in iteraplan durch die Zuordnung von *Berechtigungen* zu den jeweiligen *Anwendern* bzw. *Anwendergruppen* (vgl. [it10]).

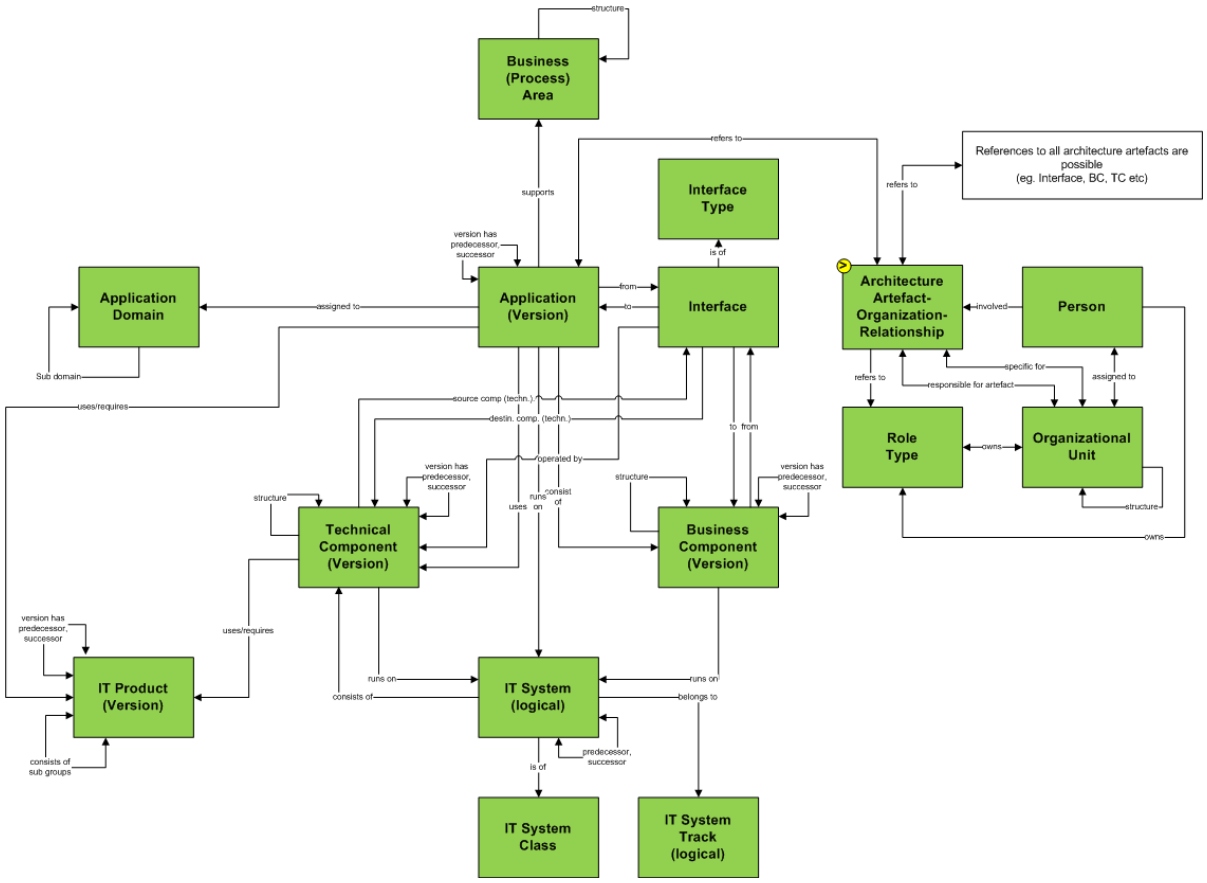

Abbildung 23 – Unternehmensarchitektur - Organisation#3
(Quelle: iteratec GmbH)

Der in Abbildung 23 dargestellte Elementtyp *Architecture Artefact-Organisation-Relationship* bildet eine Assoziationsklasse ab, welche der *Fachlichen Zuordnungen* in iteraplan entspricht. Da die von Organisation#3 definierten Beziehungen zwischen den Elementtypen, durch die Beziehungen der jeweiligen *Buildingblocktypen* abgebildet werden können, beschränken sich die durch den weiteren Vergleich der Modelle identifizierbaren Differenzen auf terminologische Unterschiede in der Bezeichnung von Elementtypen und deren Beziehungen (vgl. Tabelle 8).

4.4 Analyse unternehmensspezifischer Informationsmodellanpassungen

Nachdem im vorigen Abschnitt die theoretische Übertragbarkeit von unternehmensspezifischen EA-Informationsmodellen auf iteraplan untersucht wurden, folgt nun die Analyse von Modellierungen, welche im Zuge der Anpassung des iteraplan-Informationsmodells auf unternehmensspezifische Anforderungen durchgeführt wurden. Im Mittelpunkt stehen dabei die Ergebnisse der konkreten Modellierungen. Diese werden nach den in Abschnitt 4.1 beschriebenen Anpassungen durchsucht.

4.4.1 Untersuchte Organisationen

Die für die Analysen verwendeten Daten wurden von der iteratec GmbH in Form von anonymisierten Datensätzen einzelner iteraplan-Installationen bereitgestellt. Aus Datenschutzgründen werden die untersuchten Datensätze sowie deren erstellende Organisationen nachfolgend ausschließlich abstrakt beschrieben. Die Beschreibung der für diese Arbeit relevanten Modellierungsbeispiele erfolgt ebenfalls unabhängig vom jeweiligen Unternehmen.

Die nachfolgend aufgeführten Organisationen bzw. deren iteraplan-Datensätze, zeigen wiederum auf die unternehmensspezifischen Absichten, die durch EAM verfolgt werden sollen. Um dies zu verdeutlichen, wird den vorgestellten Unternehmen jeweils eine der in Abschnitt 2.1 beschriebenen Kategorien der EAM-Ziele (*Concerns)* zugeordnet (vgl. Tabelle 9). Hierbei kann jedoch aufgrund der vorliegenden Daten keine generelle Aussage über die Relevanz einer Kategorie getroffen werden, da diese erwartungsgemäß je nach Unternehmen variiert.

Unternehmen	EAM-Zielkategorie nach [Bu08]
Organisation#4	Project Portfolio Management
Organisation#5	Interface, Business Object, and Service Management
Organisation#6	Interface, Business Object, and Service Management
Organisation#7	Technology Homogenity; Infrastructure Management
Organisation#8	Business Process

Tabelle 9 – EAM Ziele der untersuchten Unternehmen

4.4.1.1 *Organisation#4*

Das erste Unternehmen – im Folgenden als Organisation#4 bezeichnet – tritt als Dienstleister für Logistiklösungen auf. Neben dem in iteraplan zentralen Element des Informationssystems gilt das Interesse von Organisation#4 v.a. auch der Modellierung seiner IT-Projekte und somit der Entwicklung der Informationssystemlandschaft. Dies zeigt sich nicht nur an der Anzahl an eingepflegten Instanzen, sondern auch durch die vergleichsweise hohe Zahl an *Merkmalen* zu

diesem *Buildingblocktyp* (vgl. Tabelle 10). Die vorliegenden Daten legen den Schluss nahe, dass Organisation#4 hauptsächlich EAM-Ziele der in Abschnitt 2.1 beschriebenen Kategorie „*Project Portfolio Management*“ verfolgt.

Ein weiterer wichtiger Aspekt des Informationsmodells dieses Unternehmens ist die Modellierung einer *Serviceorientierten Architektur* (SOA). Diese Modellierung wird in iteraplan aufgrund des Fehlens entsprechender *Buildingblocktypen* für Services nicht direkt unterstützt. Dies spiegelt sich im Architekturmodell durch Abbildung der EA-Konzepte Applikation und Service auf den *Buildingblocktyp Informationssystem* wieder.

Buildingblocktyp	Anzahl der Instanzen	Anzahl der Merkmale
Informationssystem	ca. 500	32
Projekt	ca. 150	25

Tabelle 10 – Datensatzstatistik Organisation#4

Das SOA-Konzept allgemein, sowie dessen mögliche Modellierung in iteraplan werden in Abschnitt 5.2.1 näher beschrieben.

4.4.1.2 *Organisation#5*

Das zweite Unternehmen – nachfolgend als Organisation#5 bezeichnet – dessen Daten auf die beschriebene Weise nach Informationsmodellanpassungen untersucht wurden, fungiert als internationaler Outsourcing-Dienstleister. Im Gegensatz zu Organisation#4, sind die wichtigsten *Buildingblocktypen* für Organisation#5 das *Informationssystem*, der *Technische Baustein* sowie die *Schnittstelle*. Auch hier wurden zur Bewertung die Anzahl an Instanzen bzw. die Anzahl an zusätzlich definierten *Merkmalen* pro *Buildingblocktyp* herangezogen (vgl. Tabelle 11). Die entsprechenden Werte deuten auf eine komplexe Informationssystemlandschaft, sowie auf die Absicht des Unternehmens diese Komplexität durch das EAM beherrschbar zu machen. Daher wird Organisation#5 die in Abschnitt 2.1 beschriebene EAM-Ziel-Kategorie des „*Interface, Business Object, and Service Management*“ zugeordnet.

Buildingblocktyp	Anzahl der Instanzen	Anzahl der Merkmale
Informationssystem	ca. 650	50
Technischer Baustein	ca. 300	20
Schnittstelle	ca. 400	10
Geschäftsobjekt	ca. 150	4

Tabelle 11 – Datensatzstatistik Organisation#5

4.4.1.3 *Oraganisation#6*

Die in Tabelle 12 dargestellte Statistik des Datensatzes eines Großkonzerns – nachfolgend als Organisation#6 bezeichnet – deutet auf komplexe Kommunikationsbeziehungen der Informationssystemlandschaft dieser Organisation hin. Daher wird Organisation#6 ebenfalls die EAM-Zielkategorie des „*Interface, Business Object, and Service Management*“ zugeordnet. Die vergleichsweise geringe Anzahl zusätzlicher *Merkmale* der entsprechenden *Buildingblocktypen* lässt sich u.a. durch den Erhebungszeitpunkt der untersuchten Daten zurückführen. Da dieser mit nur wenigen Wochen Verzögerung, vergleichsweise nahe am Zeitpunkt der iteraplan-Einführung liegt ist anzunehmen, dass die Zahl an *Merkmalen* ansteigen wird. Sobald sich das Werkzeug etabliert hat, kann mit der Entwicklung und Modellierung zusätzlicher unternehmensspezifischer EAM-Aspekte begonnen werden.

Buildingblocktyp	Anzahl der Instanzen	Anzahl der Merkmale
Schnittstelle	ca. 850	18
Informationssystem	ca. 550	9
Geschäftsobjekt	ca. 100	5

Tabelle 12 – Datensatzstatistik Organisation#6

4.4.1.4 *Organisation#7*

Tabelle 13 zeigt die Datensatzstatistik eines regionalen Energieversorgers („Organisation#7“). Dieses Unternehmen zeichnet sich besonders durch die Wichtigkeit des *Buildingblocktyps* der *Infrastrukturelemente* aus. Die hohe Anzahl an Instanzen dieses Typs deutet auf die Absicht der Organisation#7, die Standardisierung der *Betriebsinfrastruktur*- sowie der *Technischen-Architektur* mit Unterstützung des EAM-Werkzeugs zu steuern. Daher wird Organisation#7 den EAM-Zielkategorien „*Technology Homogenity*“ und „*Infrastructure Management*“ zugeordnet.

Buildingblocktyp	Anzahl der Instanzen	Anzahl der Merkmale
Technischer Baustein	ca. 1.450	7
Infrastrukturelement	ca. 1.400	7
Informationssystem	ca. 300	7

Tabelle 13 – Datensatzstatistik Organisation#7

4.4.1.5 *Organisation#8*

Das hier betrachtete Unternehmen („Organisation#8“) tritt als Dienstleister für das Outsourcing von IT-Services auf. Wie Tabelle 14 zeigt, pflegt Organisation#8 neben *Informationssystemen* hauptsächlich Elemente des Typs *Produkt*. Dies deutet auf die Verfolgung von [Bu08] beschriebenen *Concerns* der Kategorie „Business Process“ (vgl. Abschnitt 2.1).

Buildingblocktyp	Anzahl der Instanzen	Anzahl der Merkmale
Informationssystem	ca. 250	10
Produkt	ca. 250	6
Technischer Baustein	ca. 50	5
Geschäftsprozess	ca. 50	4

Tabelle 14 – Datensatzstatistik Organisation#8

4.4.2 Besondere Modellierungen

Der in Tabelle 9 dargestellte Überblick der von Unternehmen hauptsächlich verfolgten EAM-Ziele, betont die bereits beschriebene Vielfalt der Bewertung verschiedener EA-Konzepte. Eine Gemeinsamkeit der betrachteten Unternehmen (Organisationen#4-8), besteht in der Relevanz die dem *Buildingblocktyp Informationssystem* zugeschrieben wird. Dagegen kann aus den vorliegenden Daten keine generelle Aussage über die Wichtigkeit anderer *Buildingblocktypen* oder ganzer EA-Konzepte getroffen werden.

Aus Datenschutzgründen werden die in der Folge beschriebenen Anpassungen, die von den Unternehmen der vorhergehenden Abschnitte durchgeführt wurden, unabhängig von der jeweiligen Organisation beschrieben. Demgemäß werden die betrachteten Unternehmen jeweils ausschließlich als „Organisation“ bezeichnet. Sonstige Bezeichnungen, die Rückschlüsse auf die einzelnen Unternehmen erlauben, werden durch „XY“ ersetzt.

Die nachfolgend detailliert vorgestellten Anpassungen zeichnen sich durch eine Abweichung zwischen der Modellierungsabsicht der jeweiligen Organisation und dem eigentlichen Zweck der verwendeten – von iteraplan bereitgestellten – Anpassung aus. Eine derartige Anpassung besteht beispielsweise in der Definition eines zusätzlichen *Merkmals*, das eine Beziehung zwischen *Buildingblocktypen* des Informationsmodells abzubilden versucht. Eine andere Form der hier behandelten Anpassungen besteht in der Typisierung einzelner *Buildingblocktypen* durch Namenskonventionen, oder mit Hilfe eigens dafür angelegter *Aufzählungsmerkmale*.

4.4.2.1 *Abhängige Attribute*

Eine erste Modellierung die eine Diskrepanz zwischen der Modellierungsabsicht und der Möglichkeiten des in Abschnitt 3.3 beschriebenen iteraplan *Attributmechanismus* aufzeigt, bestehen in der Definition mehrerer Attribute, dessen Werte sich u.U. gegenseitig beeinflussen. Im Speziellen wurde von der Organisation beispielsweise das *Aufzählungsmerkmal* „Credit Card Information“ mit den Ausprägungen „True“ und „False“ für *Informationssysteme* definiert. Dieses *Merkmal* bildet also die Information ab, ob das entsprechende *Informationssystem* vertrauliche Kreditkarteninformationen verarbeitet. Zusätzlich pflegt die Organisation mit dem *Zahlenmerkmal* „Confidentiality“ eine Bewertung für die Vertraulichkeit der vom *Informationssystem* verarbeiteten Daten. Für die Einstufung der Vertraulichkeit werden dabei den *Informationssystemen* die Werte 1.0, 2.0 oder 3.0 zugeordnet.

Der allgemein bekannte Anspruch an eine vertrauliche Behandlung von Zahlungs- bzw. Kreditkarteninformationen, wird von der Organisation durch eine Modellierungsrichtlinie berücksichtigt. *Informationssysteme* welche nach dem Attribut „Credit Card Information“ Kreditkarteninformationen verarbeiten, besitzen demnach ausnahmslos den „Confidentiality“-Wert 3.0, und trägt so der hohen Vertraulichkeit von Kreditkarteninformationen Rechnung.

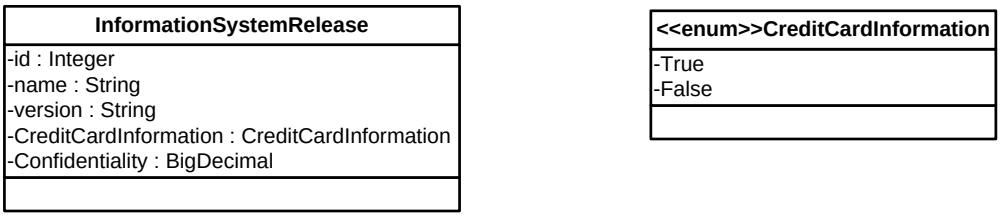

Abbildung 24 – Abhängige Attribute

Da dieser Zusammenhang jedoch nicht auf das Informationsmodell abgebildet werden kann, liegt die Verantwortung der konsistenten Pflege dieser korrelierenden Attribute, allein bei den *Anwendern*.

4.4.2.2 *Typisierung von Buildingblocktypen*

Die Organisation pflegt die Unternehmensarchitektur von zwei Geschäftsbereichen, die in iteraplan als entsprechende *Fachliche Domänen* – im Folgenden als „FD1“ bzw. „FD2“ bezeichnet - abgebildet werden. *FD2* wird dabei in der Hierarchie als Unterelement von *FD1* modelliert.

Betrachtet man nun das von der Organisation definierte *Aufzählungsmerkmal* „XY_ORG“ für den *Buildingblocktyp Fachliche Domäne*, kann der Zweck dieses *Merkmals* nicht direkt nach-

vollzogen werden. Das *Merkmal* besitzt die beiden Ausprägungen „L“ und „LC“, wobei *FD1* der Wert „LC“ und *FD2* der Wert „L“ zugeordnet wurde (vgl. Abbildung 25).

Die Tatsache, dass sich im Datensatz keine weiteren Elemente befinden, auf die diese Ausprägungen verweisen, erlaubt den Schluss, dass dieses *Merkmal* dazu bestimmt war, um die internen Strukturen des Konzerns, bestehend aus Mutterkonzern und mehreren Subunternehmen, darzustellen. Zum Untersuchungszeitpunkt, mit zwei eingepflegten *Fachlichen Domänen* sowie den zwei genannten Ausprägungen des untersuchten *Aufzählungsmerkmals*, ist die Aussagekraft dieses *Merkmals* begrenzt und lässt vermuten, dass es lediglich zur farblichen Unterscheidung von *FD1* und *FD2* bei den grafischen Auswertungen verwendet wird.

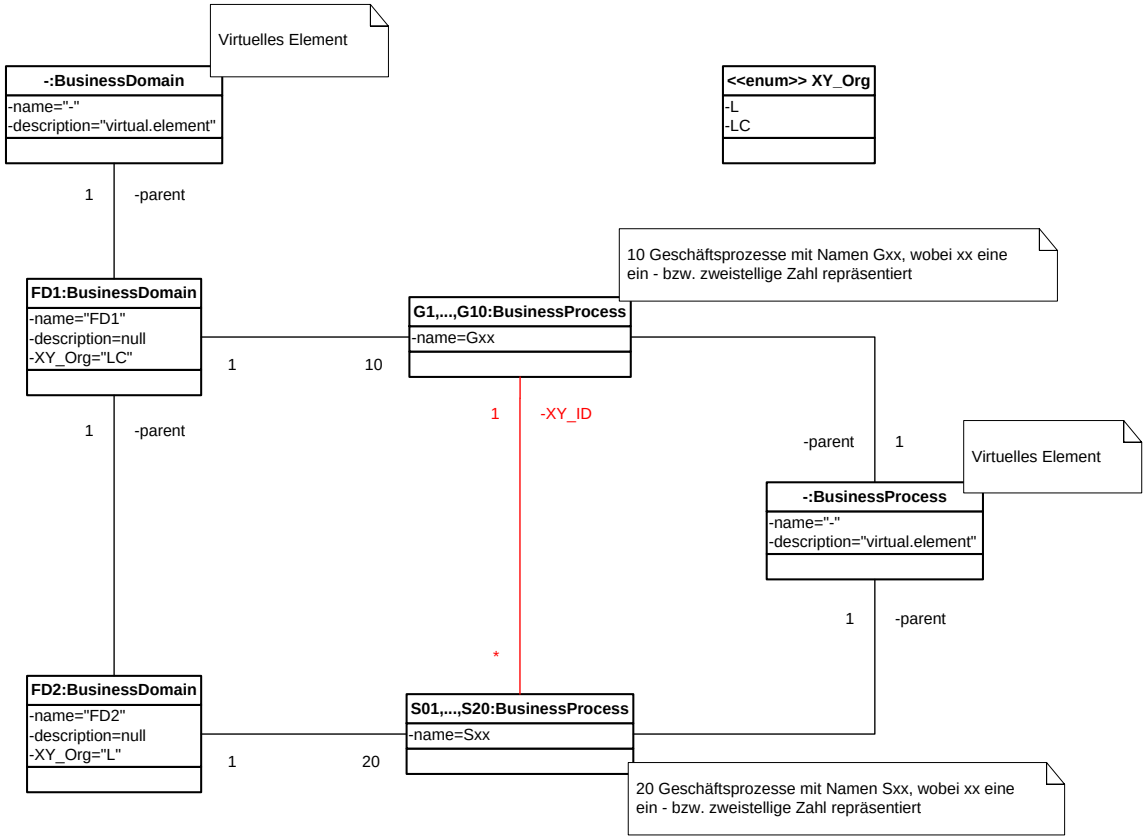

Abbildung 25 – Objektdiagramm einer SOA-Modellierung

Die Untersuchung des *Aufzählungsmerkmals „*XY_ID*“* für den *Buildingblocktyp Geschäftsprozess* stellt dagegen eine spezielle Form einer Informationsmodellerweiterung dar. Das *Merkmal* bildet eine Autoassoziation für *Geschäftsprozesse* ab. Da die Organisation sowohl seine *Geschäftsprozesse* der *Geschäftsarchitektur*, als auch die SOA-Services auf den *Buildingblocktyp Geschäftsprozess* abbildet, muss eine zusätzliche Typisierung dieses *Buildingblocktyps* eingeführt werden. Die Unterscheidung erfolgt hierbei durch Namenskonvention über dem Anfangsbuchstaben des Instanznamens. *Geschäftsprozesse* der *Geschäftsarchitektur* besitzen dabei Namen im Format „Gxx“, wobei „xx“ für ein- bzw. zweistellige Zahlen steht. *Services* werden hingegen auf *Geschäftsprozesse* mit Namen in Form von „Sxx“ abgebildet.

Hierbei steht „xx“ wiederum für eine Zahlenfolge. Neben der Typisierung des *Buildingblocktyps* bedingt die Abbildung von *Geschäftsprozessen* der *Geschäftsarchitektur* sowie der *Services* auf den gleichen *Buildingblocktyp*, die Notwendigkeit der Zuordnung von *Services* zu den *Geschäftsprozessen* der *Geschäftsarchitektur*. Hierzu wurde das *Aufzählungsmerkmal* „XY_ID“ definiert. Es ordnet mit seinen sieben Ausprägungen der Form „Gxx“ den modellierten *Services* entsprechende *Geschäftsprozesse* zu. Hierbei steht „Gxx“ wiederum für die *Geschäftsprozesse* die der *Fachlichen Domäne* „FD1“ zugeordnet wurden (vgl. Abbildung 25).

4.4.2.3 *Modellierung zusätzlicher Beziehungen*

Das für *Projekte* definierte *Freitext-Merkmal* „AffServicesAppl“ („Affected Services/Applications“) repräsentiert ein Attribut, dessen Werte auf verschiedene Elemente innerhalb des Datensatzes verweisen. Somit ordnet es *Projekten* Elemente zu, die von der entsprechenden Instanz betroffen sind (vgl. „affected“). Dies geschieht durch Wertausprägungen, die den Namen von Instanzen anderer *Buildingblocktypen* entsprechen. In diesem Sinne übernimmt das *Merkmal* eine ähnliche Funktion wie die Beziehung zwischen den *Buildingblocktypen Informationssystem* und *Projekt* („betroffene Informationssysteme“, vgl. Abschnitt 3.2). Sie erweitert diese Beziehung jedoch um die Möglichkeit, *Projekten* neben *Informationssystemen* auch Instanzen anderer *Buildingblocktypen* zuzuordnen.

Abbildung 26 – Modellierung zusätzlicher Beziehungen

Abbildung 26 zeigt die durch die Organisation gepflegte Modellierung dieser zusätzlichen Beziehung. Neben *Informationssystemen* werden durch die Wertausprägungen des betrachteten *Freitextmerkmals* zum Untersuchungszeitpunkt auch *Infrastrukturelemente*, *Geschäftsprozesse* und *Geschäftseinheiten* referenziert.

Ein weiteres Beispiel für die Modellierung zusätzlicher Beziehungen bildet das *Freitextmerkmal* „Server Name“, welches für *Informationssysteme* gepflegt werden kann. Hierzu verweisen die Merkmalswerte auf den Namen von *Infrastrukturelement*-Instanzen. Diese Modellierung ist in Abbildung 27 dargestellt.

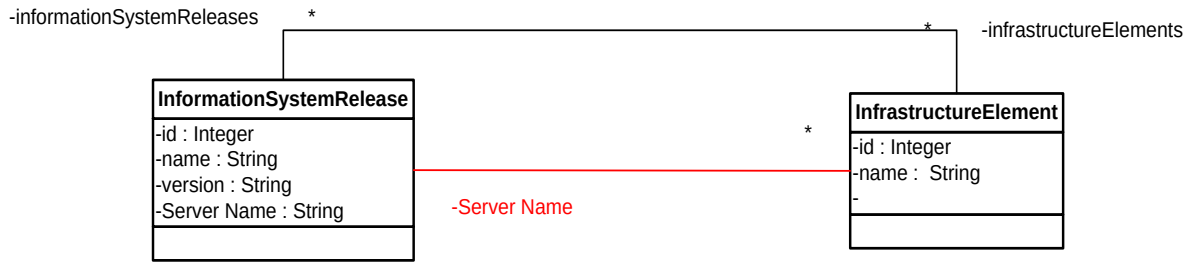

Abbildung 27 – Modellierung zusätzlicher Beziehungen II

Die hier beschriebenen Modellierungen von zusätzlichen Beziehungen bringen die Gefahr einer inkonsistenten Modellierung bzw. Datenhaltung mit sich, da es in iteraplan keinerlei Konsistenzsicherungsmaßnahmen für *Merkmale* gibt, die fremdschlüsselartig auf andere Elemente verweisen.

4.4.2.4 *Modellierung zusätzlicher Buildingblocktypen*

Das hier beschriebene Beispiel einer kundenspezifischen Anpassung zeichnet sich durch die Modellierung eines zusätzlichen *Buildingblocktyps* aus. Die Organisation versucht an dieser Stelle unternehmensspezifische Strukturen, die nach Auffassung der Modellierer nicht über *Fachliche Domänen* und *Geschäftseinheiten* abbildbar sind, zu modellieren. Im Speziellen wird hierzu ein *Freitextmerkmal* „BusinessDomain“ für Projekte angelegt. Die Ausprägungen dieses *Merkmals* verweisen auf „Geschäftsbereiche“ des Unternehmens (vgl. Abbildung 28). Da die Werte an keiner anderen Stelle des iteraplan-Datensatzes aufzufinden sind, entspricht die Pflege des *Merkmals* an dieser Stelle der Pflege eines zusätzlichen Datentyps der „Geschäftsbereiche“.

Abbildung 28 – Modellierung zusätzlicher Buildingblocktypen

Diese Modellierung zusätzlicher *Buildingblocktypen* birgt ebenfalls die Gefahr einer inkonsistenten Datenhaltung. Die Verantwortung liegt dabei allein bei den Anwendern. Die Schwierigkeit einer konsistenten Datenpflege, zeigt sich bereits anhand der zum Untersuchungszeitpunkt gepflegten Werte. So werden die Namen dieser *Geschäftsbereiche* teilweise mit verschiedenen Schreibweisen gepflegt. Die Mehrfachzuordnung der modellierten Beziehung zwischen *Projekten* und *Geschäftsbereichen* bedingt eine weitere Inkonsistenz der Datenhaltung. Dies schlägt sich in der Pflege von Werten nieder, die sich aus den Namen der entsprechenden *Geschäftsbereiche* zusammensetzen, wobei die Trennung dieser Namen ebenfalls durch unterschiedliche Zeichen dargestellt wird.

Um die Pflege sowie die Konsistenzerhaltung zu erleichtern wäre die Überlegung angebracht, das *Merkmal* durch ein *Aufzählungsmerkmal* mit zulässiger Mehrfachauswahl zu ersetzten. Dies trägt jedoch eine zusätzliche Schwierigkeit in die Informationsmodellierung ein. Die Definition und Anpassung dieses *Aufzählungsmerkmals* erfolgt auf Ebene des Informationsmodells, wohingegen die repräsentierten Konzepte, d.h. die *Geschäftsbereiche*, als Elemente in einem Architekturmodell zu verstehen wären. In diesem Sinne führt die konsistenzsichernde Modellierung mittels *Aufzählungsmerkmal* zu einer Vermischung zwischen Informationsmodell und Architekturmodell, d.h. zwischen Typ- und Instanzebene.

4.4.2.5 *Hierarchische Strukturierung*

Das *Aufzählungsmerkmal* „BU_Type“ für *Geschäftseinheiten* differenziert mit seinen beiden Ausprägungen „Own“ und „Supplier“ interne Unternehmensbereiche von externen Partnerunternehmen. Wie Abbildung 29 zeigt, ist die gewünschte Information, ob es sich bei der betrachteten *Geschäftseinheit* um eine interne oder externe Instanz handelt, bereits in der hierarchischen Zuordnung zur entsprechenden *Geschäftseinheit* „XY“ bzw. „Suppliers“ abgebildet. Da die Attribute übergeordneter Instanzen jedoch beim iteraplan Abfragemechanismus lediglich über eine Hierarchieebene hinweg berücksichtigt werden können, dient dieses *Merkmal* zur Auswahl bzw. Differenzierung der Objekte bei grafischen oder tabellarischen Auswertungen.

Dem gleichen Schema folgt das *Aufzählungsmerkmal* „Region“, das auf der dritten Hierarchieebene die geografische Differenzierung der *Geschäftseinheiten*, geordnet nach Kontinenten vornimmt (vgl. Abbildung 29).

In beiden Fällen wird die entsprechende Information mehrfach abgebildet, und es besteht die Gefahr einer inkonsistenten Modellierung. Da das Werkzeug zum Untersuchungszeitpunkt die Abbildung bestimmter Abhängigkeiten, die einen derartigen Sachverhalt modellieren, nicht unterstützt, liegt auch hier die Verantwortung einer konsistenten Datenhaltung bei den Anwendern.

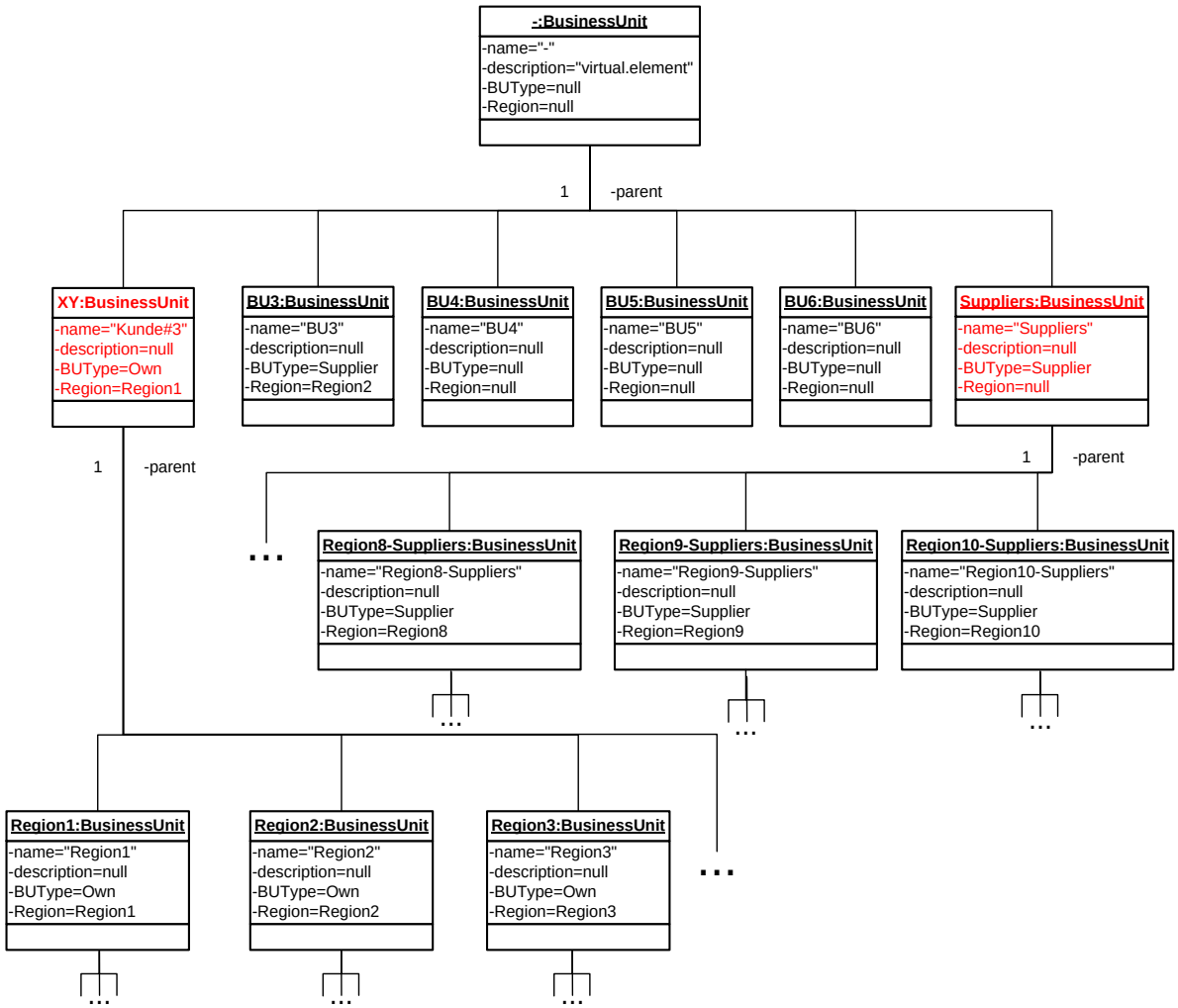

Abbildung 29 – Hierarchische Strukturierung von Geschäftseinheiten

Abbildung 29 zeigt am Beispiel der *Geschäftseinheit* „BU3“ eine bereits vorliegende inkonsistente Modellierung. Der *BU-Type*-Wert dieser Instanz deutet mit „Supplier“ zwar auf ein externes Partnerunternehmen hin, die hierarchische Ordnung wurde jedoch an dieser Stelle nicht entsprechend angepasst.

4.4.2.6 *Modellierung zusätzlicher Beziehungen für typisierte Buildingblocktypen*

Die nun betrachtete Anpassung zeichnet sich durch die Kombination mehrerer besonderer Modellierungen aus. Die Anpassung erweitert die in Abschnitt 4.4.2.5 beschriebene hierarchische Strukturierung um zusätzliche Beziehungen.

Wie Abbildung 29 zeigt, gliedert die Organisation sein Unternehmen in interne und externe Geschäftseinheiten auf der zweiten Hierarchieebene. Die dritte Ebene unterteilt die Subunternehmen weiter in verschiedene Kontinente bzw. Erdteile. Erst auf der nächsten Ebene werden die konkreten Standorte modelliert.

Die Erweiterung dieser Modellierung besteht im *Freitext-Merkmal* „Location“. Dieses *Merkmal* wurde für die *Buildingblocktypen Informationssystem* und *Projekt* angelegt. Der Name dieses *Merkmals* deutet dabei bereits auf die Modellierung eines Standorts o.Ä. hin.

Wie Abbildung 7 zeigt, existiert zwischen *Projekten* und *Informationssystemen* eine *:*-Beziehung. Diese ordnet wie bereits beschrieben *Projekten* diejenigen *Informationssysteme* zu, die von der Durchführung des *Projekts* betroffen sind. Vergleicht man nun die *Location*-Werte von *Projekten* mit den *Location*-Werten der entsprechenden *Informationssysteme,* kann dabei kein Zusammenhang zwischen den Werten festgehalten werden. Vielmehr verweisen die Ausprägungen auf verschiedene *Geschäftseinheiten* der vierten Hierarchieebene (vgl. Abbildung 29). Die Werte besitzen dabei ausnahmslos die Form „Firmenname Land“, wobei „Firmenname“ die Organisation bzw. den Namen des jeweiligen Partnerunternehmens darstellt. „Land“ verweist dagegen auf den jeweiligen Standort – meist das Land oder die Region – der *Geschäftseinheit*.

Somit besteht die resultierende Modellierung aus der Kombination einer Typisierung des *Buildingblocktyps Geschäftseinheit,* sowie einer zusätzlichen Erweiterung des iteraplan-Informationsmodells um Beziehungen (vgl. Abbildung 30). Die Typisierung erfolgt dabei in Anlehnung an die hierarchische Strukturierung der *Geschäftseinheiten* (vgl. Abbildung 29). Im Speziellen werden auf der zweiten Ebene Kontinente und auf der dritten Ebene Länder bzw. Regionen modelliert. Diese Ebenen dienen also ausschließlich zur geografischen Strukturierung der eigentlichen *Geschäftseinheiten.* Diese werden erst ab der vierten Hierarchieebene modelliert.

Die zusätzlich über das *Merkmal* „Location“ abgebildeten Beziehungen erweitern nun die konkreten *Geschäftseinheiten* um Beziehungen zu *Informationssystemen* und *Projekten*.

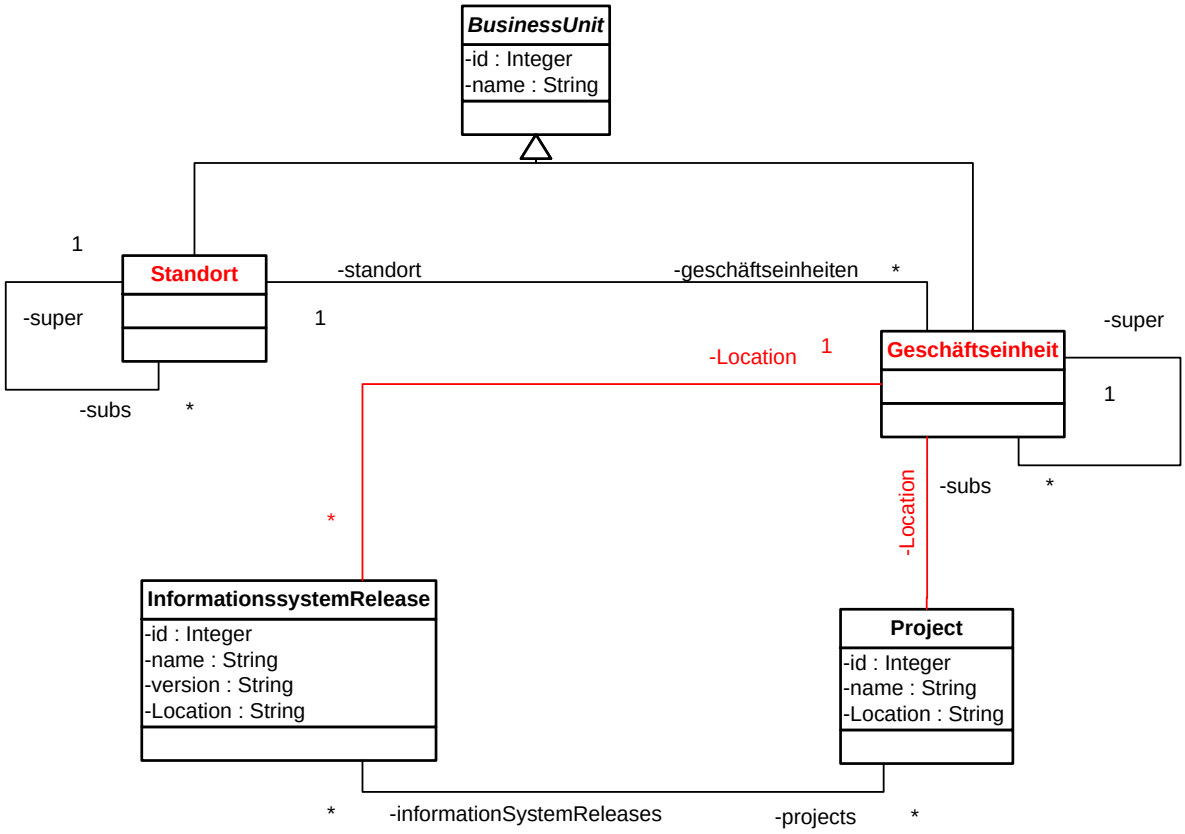

Abbildung 30 – Zusätzliche Beziehungen für typisierte Geschäftseinheiten

Abbildung 30 zeigt vereinfacht die beschriebene Modellierung. Die Klasse *BusinessUnit* bildet dabei eine abstrakte Superklasse. Zur geografischen Strukturierung wurde die Klasse *Standort* definiert. Diese bildet über die Autoassoziation „super-subs“ die Zuordnung von Ländern zu Kontinenten ab. Die Klasse *Geschäftseinheit* entspricht der ursprünglichen Modellklasse des *Buildingblocktyps Geschäftseinheit.* Wie zuvor beschrieben, werden diese über das *Merkmal Location* um Beziehungen zu den *Buildingblocktypen Informationssystem* und *Projekt* erweitert.

4.4.2.7 *Entfernung einer Beziehung*

Die letzte hier aufgeführte Anpassung, besteht in einer Informationsmodellreduzierung nach dem in Abschnitt 4.1.2 beschriebenen Anwendungsfall 2.2. Die betroffene Beziehung ist dabei die Autoassoziation von *Geschäftsobjekten*, die zur Abbildung der hierarchischen Strukturen des *Buildingblocktyps* dient. Diese Reduzierung hebt sich besonders ab, da die Organisation die Informationen der Beziehung an anderer Stelle abbildet. Hierzu wird die Hierarchie der *Geschäftsobjekte* auf bis zu vier Ebenen durch die Verwendung bestimmter Namenskonventionen modelliert. Die Namen der Objekte nehmen dabei die Form „Kategorie-Subkategorie-(Subsubkategorie-)Instanz” an. „Kategorie“, und „Sub-“ bzw. „Subsubkatego-

rie“ stehen dabei für die Namen bestimmter Kategorien. „Instanz“ bildet dann den eigentlichen Namen des jeweiligen *Geschäftsobjekts* ab.

Der Datensatz enthält ca. 60 Geschäftsobjekte, die sich auf elf *Kategorien*, welche sich wiederum in bis zu vier weitere *Subkategorien* und bis zu drei *Sub-Subkategorien* verzweigen, verteilen.

Der Grund für diese Modellierung liegt im Abfragemechanismus des Werkzeugs begründet. Dieser ermöglicht die Auswahl bestimmter Elemente des Datensatzes, indem Bedingungen für Attribute und *Merkmale* bestimmter *Buildingblocktypen* formuliert werden können (vgl. [it10]). Darüber hinaus können auch Bedingungen für direkt referenzierte Instanzen des gleichen sowie anderen *Buildingblocktypen* formuliert werden. Dies erlaubt zwar die Abfrage über eine Hierarchieebene, eine Ausweitung über mehrere Stufen ist dagegen nicht möglich.

Durch die Abbildung der hierarchischen Struktur über die beschriebenen Namenskonventionen können so über den Abfragemechanismus Elemente aller Hierarchieebenen zusammengefasst bzw. ausgewählt werden.

Auch hierbei besteht die Gefahr einer inkonsistenten Modellierung, und die Verantwortung einer konsistenten Datenhaltung liegt erneut ausschließlich bei den Anwendern. Fehler in der Benennung bestimmter Elemente können von iteraplan nicht als solche identifiziert bzw. gekennzeichnet werden.

4.5 Aktuelle Modellierungsanforderungen

Neben den Untersuchungen der von den Organisationen erstellten EA-Informationsmodellen, sowie den Anpassungen an dem von iteraplan vordefinierten Informationsmodell, werden abschließend zusätzlich Anforderungen beschrieben, die in aktuellen EAM-Beratungsprojekten der iteratec GmbH erhoben wurden. Für den Gegenstand vorliegender Arbeit wurden diese Anforderungen während eines Interviews durch den iteraplan Produktmanager Karsten Voges formuliert.

4.5.1 Typisierung von Projekten

Da nicht alle Änderungsvorschläge an die Anwendungslandschaft direkt in IT-Änderungsprojekte (vgl. *Buildingblocktyp Projekt*) fließen, besteht eine Anforderung aus der Modellierung von *Änderungsanforderungen*, die u.U. in *Projekte* umgewandelt werden (vgl. Abbildung 31). Hierbei betreffen sowohl *Änderungsanforderungen* als auch *Projekte* bestimmte *Informationssysteme*.

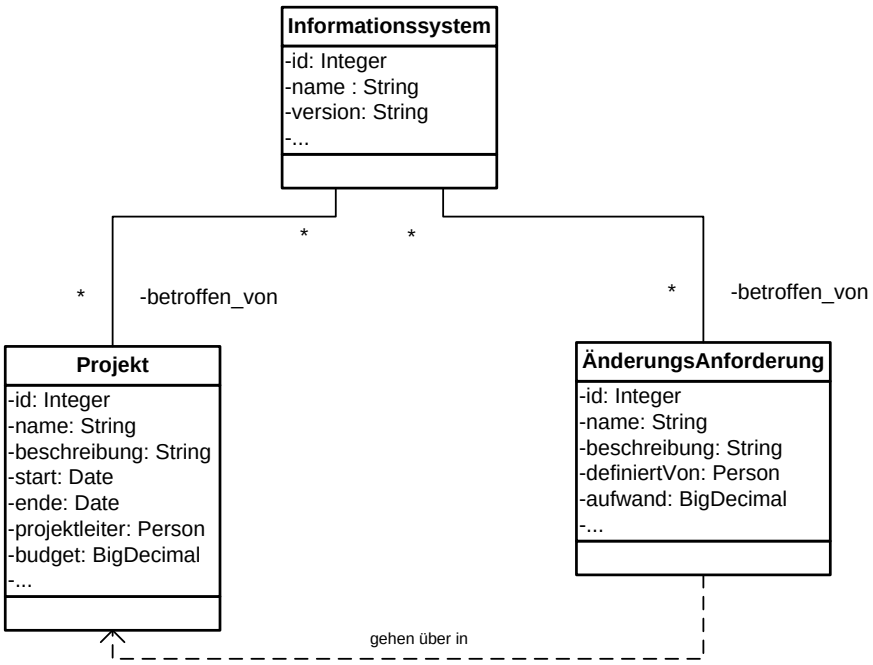

Abbildung 31 – Entwicklung von Projekten aus Änderungsanforderungen

Da iteraplan keinen *Buildingblocktypen* bereitstellt, der die Klasse *ÄnderungsAnforderung* direkt abbildet und keine zusätzlichen *Buildingblocktypen* definiert werden können, besteht eine mögliche Modellierung in iteraplan, in der zusätzlichen Differenzierung des *Buildingblocktyps Projekt*. Für die Differenzierung stehen in iteraplan grundsätzlich zwei Alternativen zur Verfügung. Die erste Möglichkeit ist die Definition eines *Aufzählungsmerkmals* „Typ“, sowie Wertausprägungen „Projekt“ und „Anforderung“. Dieses Attribut dient somit als *Diskriminator* und zeigt mit der Ausprägung auf den jeweiligen Subtyp. Die zweite Alternative besteht in

der Verwendung von Namenskonventionen. So könnten beispielsweise Änderungsanforderungen bei ihrer Benennung, durch die Verwendung des Prä- bzw. Suffixes „Anforderung“ als solche gekennzeichnet werden. Bei einer etwaigen Übertragung der *ÄnderungsAnforderung* in ein *Projekt*, könnten dann die entsprechenden Zusätze aus dem Namen entfernt werden, bzw. neue *Projekte* mit gleichem Namen ohne diesen Prä- bzw. Suffix erstellt werden. Dies brächte den Vorteil einer zusätzlichen Abbildbarkeit der Information, aus welchen *Änderungsanforderungen* die einzelnen *Projekte* hervorgingen. Diese Informationen könnten durch die Verwendung der von iteraplan definierten Autoassoziation zur Abbildung der Hierarchie von *Projekten* modelliert werden. Der Nachteil all dieser Möglichkeiten liegt jedoch in der eingeschränkten Möglichkeit, Konsistenzbedingungen bezüglich Namenskonventionen bzw. Zustandsübergänge abzubilden (vgl. Tabelle 15). Diese können zwar vom jeweiligen Unternehmen formuliert, jedoch vom Werkzeug nicht automatisch überprüft werden.

	Alternative 1	**Alternative 2**
	Differenzierung erfolgt über Diskriminatorattribut (*Aufzählungsmerkmal*)	Differenzierung über Namenskonventionen (Prä- bzw. Suffix)
Vorteile	Über die Kennzeichnung des Attributs als „Pflichtmerkmal“ wird die Konsistenz der Differenzierung unterstützt	Intuitive Verbindung von Änderungsanforderungen und entsprechenden Projekten, sowie die Möglichkeit eine Instanz für beide „Typen“ anzulegen
Nachteile	Da nicht mehrere Projekte mit gleichem Namen, gleichzeitig existieren dürfen, kann die Änderungsanforderung nach einem Zustandsübergang nicht bestehen bleiben	Verantwortung einer konsistenten Modellierung liegt allein bei den Anwendern

Tabelle 15 – Modellierungsmöglichkeiten für Änderungsanforderungen

Abbildung 32 zeigt die für iteraplan vorgeschlagene Modellierung der Problemstellung von *Projekten*, die sich aus Änderungsanforderungen entwickeln. Hier wird eine Lösung beschrieben, die sowohl versucht den Vorteil einer Definition eines Pflicht-*Aufzählungsmerkmals*, als auch die Abbildung des Zusammenhangs von *Projekten* sowie den diesen zugrunde liegenden *Änderungsanforderungen* zu verbinden. Die Kennzeichnung des *Diskriminatormerkmals* als Pflichtmerkmal ermöglicht die automatische Identifikation von Instanzen ohne entsprechenden Wert. Die Verwendung der Hierarchiebeziehung ermöglicht es zu *Projekten*, diejenigen Änderungsanforderungen zuzuordnen, aus denen sie hervorgingen. Für die Sicherstellung einer ausschließlichen Zuordnung von *Änderungsanforderungen* als Nachkommen der *Projekte*, sind dabei jedoch alleine die Anwender verantwortlich.

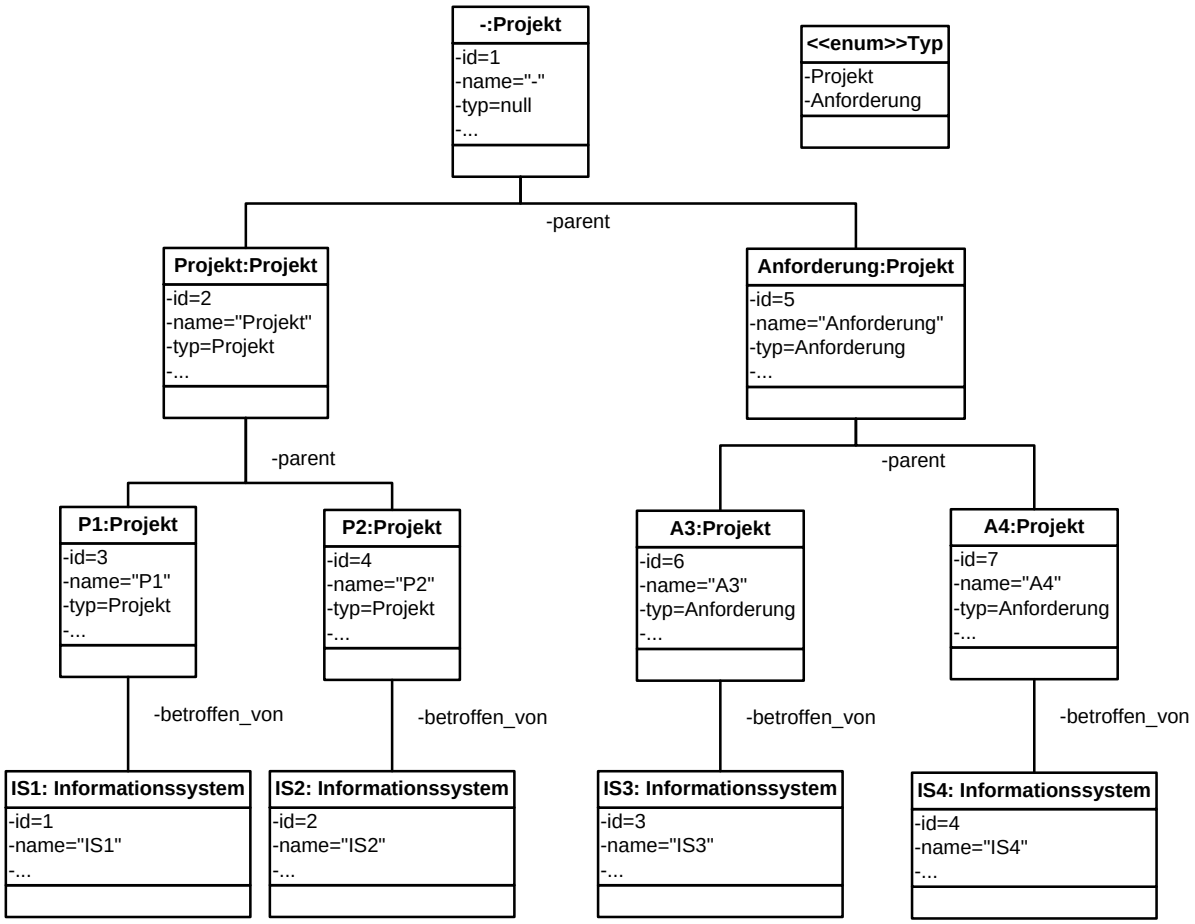

Abbildung 32 – Änderungsanforderungen – Modellierung in iteraplan

Zusätzlich zur beschriebenen Differenzierung sollten auch Attribute zur weiteren Beschreibung der jeweiligen Subtypen definiert werden. Als Beispiel hierfür können die in Abbildung 31 eingeführten Attribute herangezogen werden. Diese können durch den iteraplan *Attributmechanismus* für den *Buildingblocktyp Projekt* modelliert werden. Die Verantwortung der konsistenten Pflege dieser Attribute bzw. *Merkmale* liegt auch hier ausschließlich in der Verantwortung der Anwender. Selbst bei der Einhaltung aller Konsistenzrichtlinien, hat eine solche Typisierung den Nachteil einer Vielzahl an NULL-Werten. Diese NULL-Werte sollten dabei beispielsweise für jedes *Änderungsanforderungs*-spezifische *Merkmal* einer *Projekt*-Instanz auftreten.

4.5.2 Informationssystemdomänen

Eine weitere Modellierungsanforderung konnte im Beratungsprojekt einer weiteren Organisation identifiziert werden. Diese betrifft dabei die Gruppierung von *Informationssystemen*, welche nach dem Verständnis des Unternehmens grundsätzlich in „technische“ sowie „fachliche“ *Informationssysteme* differenziert werden können.

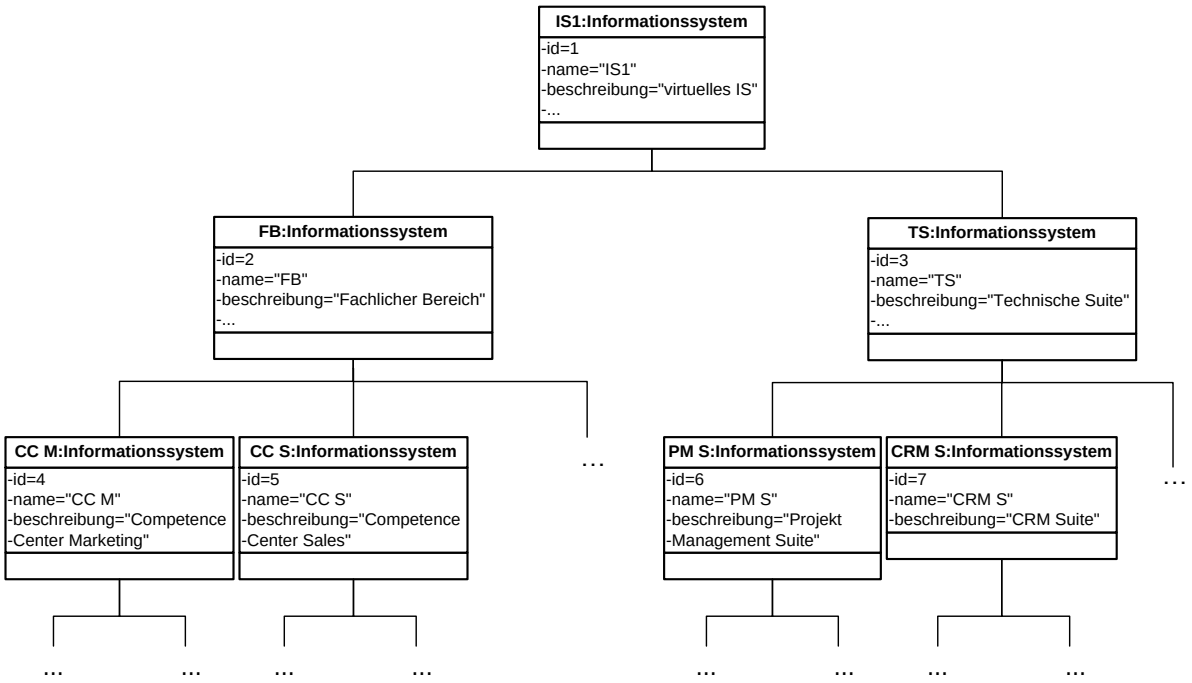

Abbildung 33 – Informationssystemgruppierung

Wie Abbildung 33 zeigt, verwendet die Organisation hierzu eine hierarchische Strukturierung der *Informationssysteme*. Auf Höhe der zweiten Hierarchieebene wird die Differenzierung zwischen *fachlichen* und *technischen Informationssystemen* durch die abstrakten Instanzen „*FB*“ („Fachlicher Bereich) bzw. „TS“ („Technische Suite“), vorgenommen. Auf der dritten Hierarchieebene untergliedert sich *FB* weiter in die fachlichen Bereiche des Unternehmens wie z.B. „CC M“ („Org-Bereich Marketing“) und „CC S“ („Org-Bereich Sales“). Parallel kann *TS* weiter in einzelne übergeordnete Querschnittsaufgaben untergliedert werden. Als Beispiel hierfür sind *Informationssysteme* zur Unterstützung des Projektmanagements, in der abstrakten Instanz „PM S“ („Projektmanagement-Suite“) zusammengefasst. Ab der dritten Hierarchieebene werden dann die konkreten *Informationssystem*-Instanzen modelliert.

Obwohl iteraplan, wie bereits in Abschnitt 3.2 beschrieben, *Informationssysteme* zur Strukturierung hierarchisch anordnen kann, verwendet die in Abbildung 34 vorgeschlagene Modellierung der Organisationsstrukturen des Sportartikelherstellers in iteraplan, den dafür bereitgestellten *Buildingblocktyp* der *Informationssystemdomänen*.

	Modellierung I	Modellierung II
Beschreibung	IS-Gruppierung über hierarchische Strukturierung der Informationssysteme	IS-Gruppierung über Informationssystemdomänen
Vorteile	-	Vereinfachte Gruppierungsmöglichkeit durch Verwendung eines speziell dafür vorgesehenen Buildingblocktyps
Nachteile	Hoher Anteil an abstrakten Informationssystemen. Gruppierung könnte bei den von iteraplan unterstützten Auswertungsmöglichkeiten nur bedingt berücksichtigt werden	-

Tabelle 16 – Modellierungsmöglichkeiten für die IS-Gruppierung

Diese können entsprechend der in Abbildung 33 dargestellten „FB“ bzw. „TS“, sowie deren direkte Nachfolger, angelegt und gepflegt werden. Die Differenzierung von *fachlichen* bzw. *technischen Informationssystemen* erfolgt hier auf der zweiten Hierarchieebene, durch die entsprechenden *Informationssystemdomänen* „FB“ sowie „TS“. Auch die Informationssysteme der dritten Hierarchieebene der Modellierung in Abbildung 33, werden in iteraplan durch die Bildung der entsprechenden Informationssystemdomänen modelliert. Die Vor- bzw. Nachteile der beiden Modellierungen des beschriebenen Sachverhalts wurden in Tabelle 16 gegenübergestellt.

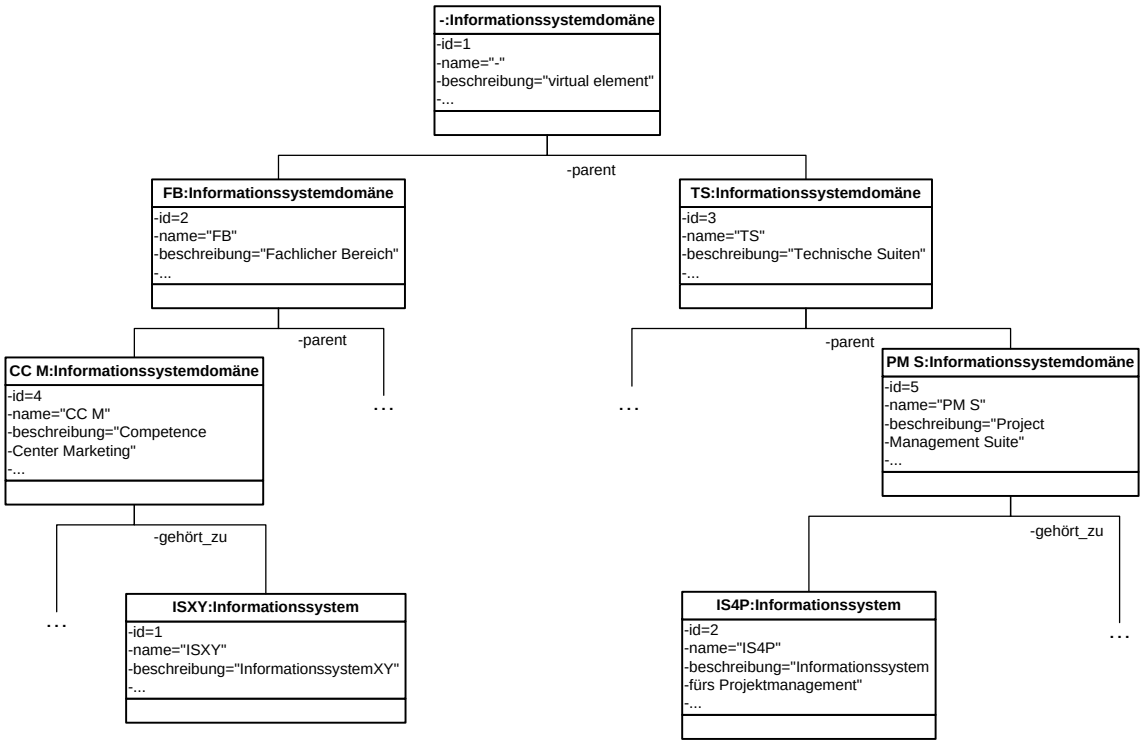

Abbildung 34 – Informationssystemgruppierung in iteraplan

4.5.3 Test-Umgebung

Im Zuge einer Reihe von Beratungsprojekten stellte sich die Wichtigkeit einer Modellierung des Entwicklungsstatus von *Informationssystemen* heraus. Dieser durchläuft, wie bereits in Abbildung 16 dargestellt, eine Reihe vordefinierter Zustände von Beginn der Entwicklung (vgl. „spezifiziert“) bis hin zur Inbetriebnahme (vgl. „aktiv“). Eine sich häufig wiederholende Modellierungsanforderung besteht daher in der Abbildung von *Informationssystemen* in verschiedenen „Umgebungen“. Die am häufigsten berücksichtigten Ausprägungen dieser *Umgebung* sind dabei „Test“-, „Integrations“- sowie „Produktivumgebung“. Abbildung 35 stellt diesen Sachverhalt beispielhaft dar. Sie zeigt *Informationssysteme* für die ein Statusattribut („zustand") zur Differenzierung der jeweiligen *Umgebung* definiert wurde. Die *Informationssysteme* werden zusätzlich *Test-*, *Integrations-*, bzw. *Produktivservern* zugeordnet, welche als *Infrastrukturelement* modelliert werden. Zur Differenzierung ihres Verwendungszwecks wurde zusätzlich ein entsprechendes *Aufzählungsmerkmal* für *Infrastrukturelemente* definiert. Dieses verweist mit den Ausprägungen „Test“, „Integration“ sowie „Betrieb“ auf die jeweilige Verwendung bzw. *Umgebung*.

Dieser Modellierungsansatz entspricht einer möglichen Lösung des beschriebenen Sachverhalts in iteraplan. Somit bietet der iteraplan *Attributmechanismus* eine ausreichend große Aussagekraft zur Modellierung einer *orthogonalen Typisierung*. Jedoch können auch hier keinerlei konsistenzerhaltende Regeln definiert bzw. automatisch überprüft werden. Daher liegt beispielsweise die Verantwortung einer einheitlichen Zuordnung von *Informationssystemen* zu *Infrastrukturelementen* der entsprechenden *Umgebung*, ausschließlich bei den *Anwendern*.

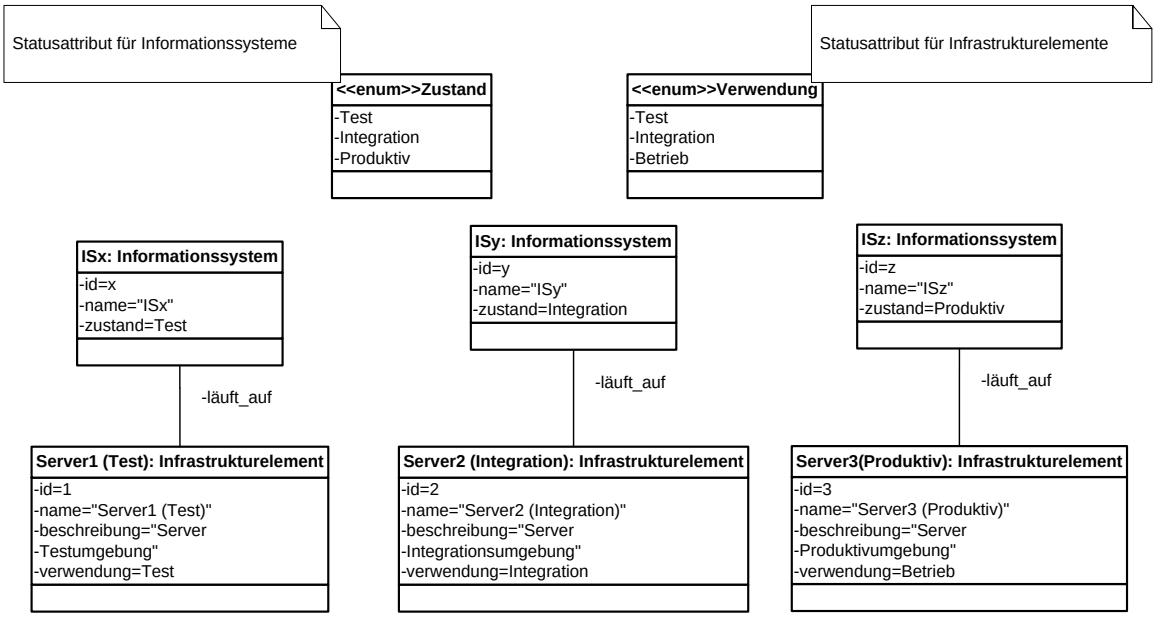

Abbildung 35 – Testumgebung

5 Konsolidierung von Anforderungen

Nachdem im vorherigen Abschnitt sowohl vordefinierte EA-Informationsmodelle verschiedener Unternehmen auf ihre Übertragbarkeit in das iteraplan-Informationsmodell, als auch die Informationsmodellanpassungen einzelner Unternehmen in iteraplan beschrieben wurden, können nun daraus resultierende Änderungsanforderungen, an die von iteraplan bereitgestellten Anpassungsmechanismen, abgeleitet werden. Neben dem in Abschnitt 3.3 beschriebenen *Attributmechanismus* betrifft dies auch das EA-Framework der iteratec GmbH, welches dem untersuchten Werkzeug zugrunde liegt. Die nachfolgende Beschreibung dieser neu erhobenen Anforderungen wird zusätzlich um abstrakte Lösungen ergänzt, welche die jeweilige Anforderung umsetzt.

5.1 Informationsmodell

Die Ergebnisse aus Kapitel 3.4 zeigen den Bedarf zusätzlicher Anpassungsmöglichkeiten des Informationsmodells auf, welche zum Untersuchungszeitpunkt nicht vom Werkzeug unterstützt werden. Die nachfolgend beschriebenen Anforderungen zielen dabei zwar auf zusätzliche Modellierungsmöglichkeiten des iteraplan-Informationsmodells ab, doch können diese auf Änderungsanforderungen des *Attributmechanismus* bzw. dessen Informationsmodell (vgl. Abbildung 8) übertragen werden.

5.1.1 Abhängige Attribute

Die in Abschnitt 4.4.2.1 beschriebene Abhängigkeit zwischen den Werten verschiedener *Merkmale* deutet auf den Bedarf, derartige Abhängigkeiten definieren und auf das iteraplan-Informationsmodell abbilden zu können.

Im Speziellen sollte es demnach in iteraplan möglich sein, Bedingungen der Form „*wenn - dann*“ zu formulieren. Im Fall der in Abschnitt 4.4.2.1 beschriebenen Abhängigkeit, sähe eine derartige Bedingung beispielsweise wie folgt aus:

> *Wenn:* Informationssystem.Credit Card Information == „True“
> *Dann:* Informationssystem.Confidentitality >= 3

Hierbei ist zu beachten, dass die *Merkmale* nicht zwangsläufig vom gleichen *Merkmalstyp* sein müssen. Vielmehr müssen die verwendeten Komparatoren auf den jeweiligen Typ des in der *Wenn*- bzw. *Dann*-Komponente verwendeten *Merkmals* abgestimmt werden. Tabelle 17

zeigt hierfür eine mögliche Zuordnung von Komparatoren zu den entsprechenden *Merkmalstypen*. Mit diesen Mengen an merkmalstypspezifischen Komparatoren wäre es möglich *Bedingungen* zu formulieren, mit Hilfe derer eine Einschränkung möglicher Wertausprägungen vorgenommen werden kann. Die Zuordnung der Komparatoren zu den entsprechenden *Merkmalstypen* ist dabei an den iteraplan *Abfragemechanismus* angelehnt (vgl. [it10]). Die Komparatoren ermöglichen die Prüfung auf *Gleichheit* („ist"), *Ungleichheit* („ist nicht"), *Null*(„kein Wert") sowie *not-Null* („beliebiger Wert") für jeden *Merkmalstyp*. Die Menge der Komparatoren wurde – verglichen mit dem *Abfragemechanismus* des Werkezugs – lediglich für *Aufzählungsmerkmale* erweitert. Die Erweiterung besteht dabei in den Komparatoren „kleiner(-gleich)" und „größer(-gleich)". Die Verwendung dieser zusätzlichen Komparatoren setzt jedoch eine Erweiterung des entsprechenden *Merkmalstyps* voraus, durch die eine sortierte Liste der möglichen Ausprägungen des *Merkmals* gepflegt werden kann.

Merkmalstypen	**Komparatoren**
Alle Merkmalstypen	*ist, ist nicht, kein Wert, beliebiger Wert*
Aufzählungs-Merkmal	*kleiner(-gleich), größer(-gleich), enthält (nicht), fängt (nicht) an mit, hört (nicht) auf mit*
Datums-Merkmal	*am, vor, nach*
Freitext-Merkmal	*enthält (nicht), fängt (nicht) an mit, hört (nicht) auf mit*
Verantwortlichkeits-Merkmal	*enthält (nicht), fängt (nicht) an mit, hört (nicht) auf mit*
Zahl-Merkmal	*kleiner(-gleich), größer(-gleich)*

Tabelle 17 – Komparatoren
(In Anlehnung an [it10])

Die beschriebene Modellierung von Abhängigkeiten verschiedener *Merkmale*, bei denen Wertausprägungen eines *Merkmals*, die zulässigen Werte eines zweiten *Merkmals* bedingen, können neben diesem einfachen Fall auch andere Anforderungen abdecken.

Eine weitere Anwendung einer solchen Abhängigkeitsmodellierung, besteht in der Pflege von „Qualitätsattributen", welche nach [Bu09] (vgl. „quality attributes") zu einer wichtigen Erweiterung von EA-Informationsmodellen zählen. Ein typisches Beispiel hierfür besteht in Anlehnung an [Bu09] aus der Pflege eines Attributs „Verfügbarkeit" (vgl. „availability") für *Informationssysteme* und *Infrastrukturelemente*. Dieses Attribut kann dabei eine beliebige Metrik abbilden. Eine beispielhafte Abhängigkeit besteht in diesem Fall darin, dass die *Verfügbarkeit* eines *Informationssystems* nicht höher bewertbar sein sollte, als die *Infrastrukturelemente*, auf denen die jeweilige Instanz betrieben wird. Dieses Beispiel lässt sich dabei nach [Bu09] auf eine Reihe weiterer *Qualitätsattribute* übertragen, was die Wichtigkeit dieser Modellierung für EA-Informationsmodelle zusätzlich betont.

Abbildung 36 stellt auf hohem Abstraktionsniveau eine mögliche Umsetzung eines Bedingungskonzepts dar. Diese Lösung würde es bereits ermöglichen, einfache *Bedingungen* für die zulässigen Werte eines *Merkmalstyps* zu definieren. Die vorgeschlagene Erweiterung führt hierzu eine Klasse *Bedingung* ein, welche neben den Standardattributen „*id*“, „*name*“ und „beschreibung“ weitere Attribute zur Abbildung von *Wenn-Dann-Bedingungen* enthält. Die *Bedingung* kann dadurch wie folgt gebildet werden:

Wenn: wenn_AT.value wenn_komparator wenn_vergleichswert.value
Dann: value dann_komparator dann_vergleichswert.value

So könnte bei der Pflege von *Merkmalen* denen *Bedingungen* zugeordnet sind überprüft werden, ob der zugeordnete Vergleichswert („wenn_AT.value“) die *Wenn*-Komponente erfüllt. Wenn dies zutrifft ist sicherzustellen, dass die *Dann*-Komponente ebenfalls erfüllt wird. Bei einer Verletzung der *Bedingung* bestünde die Möglichkeit, den verletzenden Wert als solchen zu kennzeichnen oder den Wert abzulehnen.

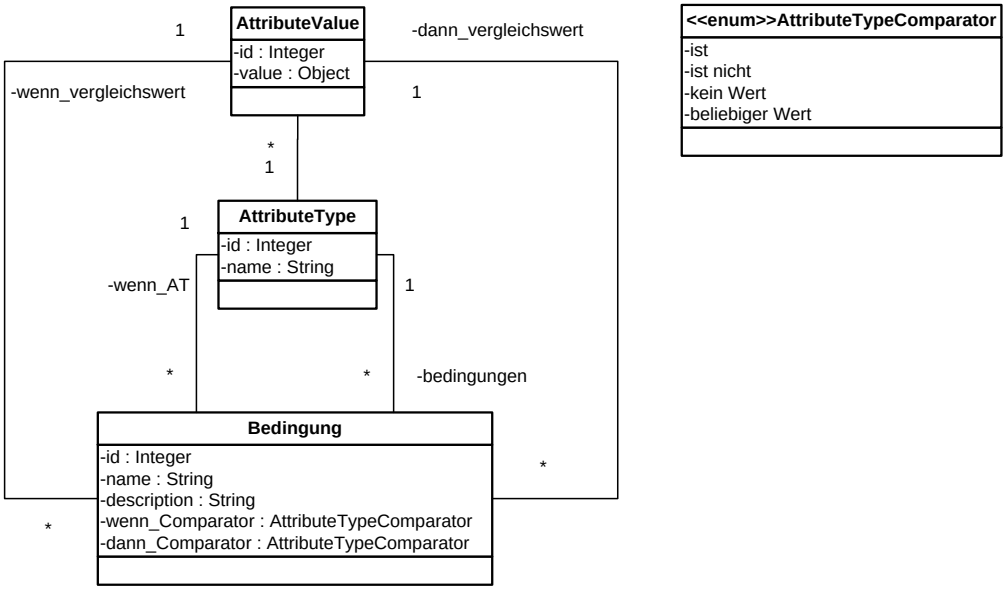

Abbildung 36 - Modellierung von Bedingungen

Um die Aussagekraft und somit gleichzeitig die Nützlichkeit dieses Bedingungsmechanismus zu erhöhen, wäre eine Erweiterung der Funktionalität um die Möglichkeit der Kombination verschiedener *Bedingungen* sinnvoll. Denkbare Kombinationsmöglichkeiten bestehen dabei in der Abbildung der boolschen Operatoren „und“ bzw. „oder“. Zusätzlich sollte es möglich sein, die Überprüfung von *Merkmalstypen* (vgl. *AttributeType*) des eigenen *Buildingblocktyps*, auf Standardattribute sowie auf *Merkmale* und Standardattribute referenzierter Objekte auszuweiten.

5.1.1.1 *Abhängige Dropdownauswahl*

Die nun betrachteten Anwendungsfälle stehen in engem Zusammenhang mit den soeben beschriebenen abhängigen Attributen. Ein typischer und allgemein bekannter Vertreter dieser Anwendungsfälle tritt häufig in Adressformularen verschiedener Internetseiten auf: Hierbei werden die Benutzer dazu aufgefordert aus einer *Dropdownauswahl* nacheinander „Land“, „Bundesland“ sowie die „Stadt“ anzugeben. Hierbei werden natürlich die zur Verfügung stehenden Bundesländer je nach Auswahl des Landes vorselektiert. Anwendungsfälle dieser Art, könnten ebenfalls mit der vorgeschlagenen Erweiterung des *Attributmechanismus* um *Bedingungen*, abgedeckt werden (vgl. Abschnitt 5.1.1).

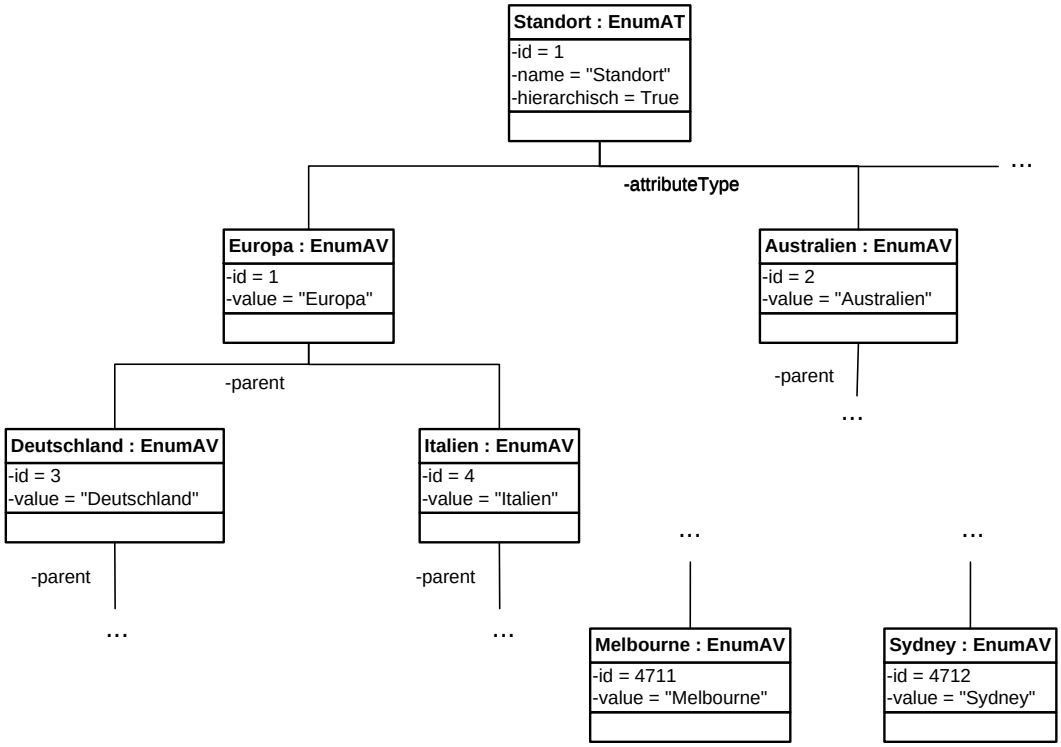

Abbildung 37 – Objektdiagramm abhängiger Merkmalswerte

Eine weitere einfache Möglichkeit den Anwendungsfall von abhängigen *Dropdownauswahlen* umzusetzen, besteht in einer hierarchischen Anordnung von Werten bestimmter *Aufzählungsmerkmale* (vgl. Abbildung 37). Dies wird durch die Definition einer zusätzlichen Autoassoziation der Kardinalität 1:* in der Klasse *EnumAV* (vgl. „parent“-Zuordnung) ermöglicht. Unter Verwendung dieser Beziehung können so beliebig verschachtelte Wertebenen angelegt und gepflegt werden. Zusätzlich kann zur Unterscheidung von konventionellen und abhängigen *Aufzählungsmerkmalen* ein Boolean-Attribut „hierarchisch“ in der Klasse *EnumAT* angelegt werden.

Abbildung 37 zeigt beispielhaft die Modellierung einer geografischen Strukturierung von Standorten in *Kontinente*, *Länder* und *Städte*. Aus Übersichtlichkeitsgründen wurde die Zu-

ordnung der Attributwerte (vgl. *EnumAV*) zum *Merkmal* „Standort“ nur auf der Ebene der *Kontinente* vorgenommen.

Diese Möglichkeit, die Pflege abhängiger *Dropdownauswahlen* zu ermöglichen, beschreibt eine, verglichen mit der Implementierung des in Abschnitt 5.1.1 vorgestellten *Bedingungs*-Mechanismus, einfach zu realisierende Umsetzung dieser Anforderung.

5.1.1.2 *Anwenderbezogene Wertpflege*

Eine weitere denkbare Erweiterung des iteraplan *Attributmechanismus* besteht in der Möglichkeit, zusätzliche *Merkmalstypen* einzuführen, deren eingepflegte Attributwerte den jeweiligen *Anwendern* zugeordnet werden. Somit können die dargestellten Werte einzelner *Merkmalstypen* durch eine Vielzahl von Werten beschrieben werden. So könnten beispielsweise *Zahlenmerkmale* definiert werden, deren dargestellter Wert dem Durchschnitt aller abgegebenen Bewertungen entspricht.
Eine Implementierung, welche diesen Sachverhalt ermöglicht wäre die Erweiterung des in Abbildung 8 dargestellten *Attributmechanismus* in folgender Form:

- Änderung der Kardinalität der Beziehung zwischen den Klassen *AttributeValue* und *AttributeValueAssignment* von 1:* zu *:*.
- Definition einer zusätzlichen Beziehung zwischen den Klassen *AttributeValue* zu *UserEntity*, die jeder *AttributeValue*-Instanz den Benutzer zuordnet von dem der Wert angelegt wurde.
- Definition eines zusätzlichen Boolean-Attributs „anwenderspezifisch“, welche zur Differenzierung zwischen klassischen und den erweiterten *anwenderspezifischen Merkmalen* dient.

Somit könnten die dargestellten Werte unter Verwendung bestimmter Funktionen aus den verschiedenen Werten berechnet werden. Beispiele für derartige Funktionen die auf *Zahlen-Merkmale* angewendet werden können sind dabei *Durchschnitt, Maximum oder Minimum*. So könnten bereits in Abschnitt 5.1.1 beschriebene EAM-typische *Qualitätsmerkmale*, unter Beachtung mehrerer Bewertungen bestimmt werden. Ein Beispiel hierfür besteht in Anlehnung an [it10] in einer diskreten Bewertung des „Gesundheitszustands“ verschiedener *Buildingblocktypen*. Dieser könnte durch die beschriebene Erweiterung um *anwenderspezifische Merkmale* die Bewertungen verschiedener Anwender während der Datenpflege berücksichtigen. Somit können u.U. Entscheidungsfindungsprozesse über derartige Bewertungen durch das Werkzeug unterstützt werden.

Durch die zusätzliche Erweiterung der Klasse *AttributeValue* um ein Integer-Attribut „gewichtung“ können die *anwenderspezifischen Merkmalswerte* beispielsweise in Abhängigkeit der *Rolle* des jeweiligen *Anwenders* zusätzlich bewertet werden (vgl. Abbildung 38).

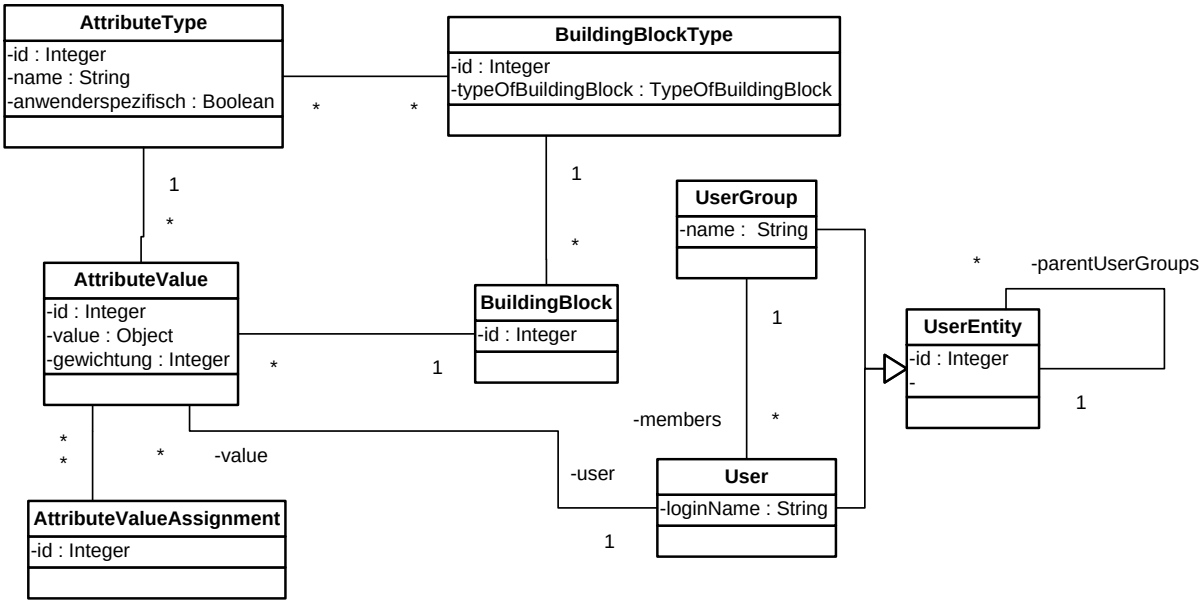

Abbildung 38 - Anwenderspezifische Merkmalstypen

Ein weiterer Anwendungsfall, der durch eine zusätzliche Erweiterung ermöglicht wird, besteht in der anwenderspezifischen Pflege einzelner *Merkmalswerte*. So könnten nach der Zuordnung von *Merkmalswerten* zu ihren erzeugenden *Anwendern*, auch *Rechte* definiert werden. Dies würde dazu führen, dass Lese- bzw. Schreibrechte der jeweiligen Werte, abhängig für einzelne *Benutzer* verwaltet werden können. Ein denkbarer Anwendungsfall für derartige *Merkmalstypen* besteht in *Textmerkmalen* zur Kommentierung einzelner Elemente. Hier wäre es möglich die Rechteverwaltung so vorzunehmen, dass alle Kommentare zwar lesbar sind, jedoch nur eigens Verfasste verändert werden können.

5.1.2 Beziehungen

Wie eine Reihe der Untersuchungen der in Kapitel 4.4.2 beschriebenen Modellierungen zeigen, besteht ein Bedarf bei der Anpassung des von iteraplan bereitgestellten Informationsmodells, in der Definition zusätzlicher Beziehungen (vgl.4.4.2.3 und 4.4.2.6) sowie Autoassoziationen einzelner *Buildingblocktypen* (vgl. Abbildung 25). Ein möglicher Ansatz, welcher dies umsetzt, wird durch den in Abbildung 8 vorgestellten *Attributmechanismus* vorgestellt. Da *Verantwortlichkeitsmerkmale* bzw. die diesen *Merkmalstyp* repräsentierende Klasse *ResponsibilityAT*, bereits eine Beziehung zwischen den jeweiligen *Buildingblocktypen* zu den *Anwendern* (vgl. Klasse *UserEntity)* darstellen, könnte dieses Konzept auch auf andere *Buildingblocktypen* übertragen werden.

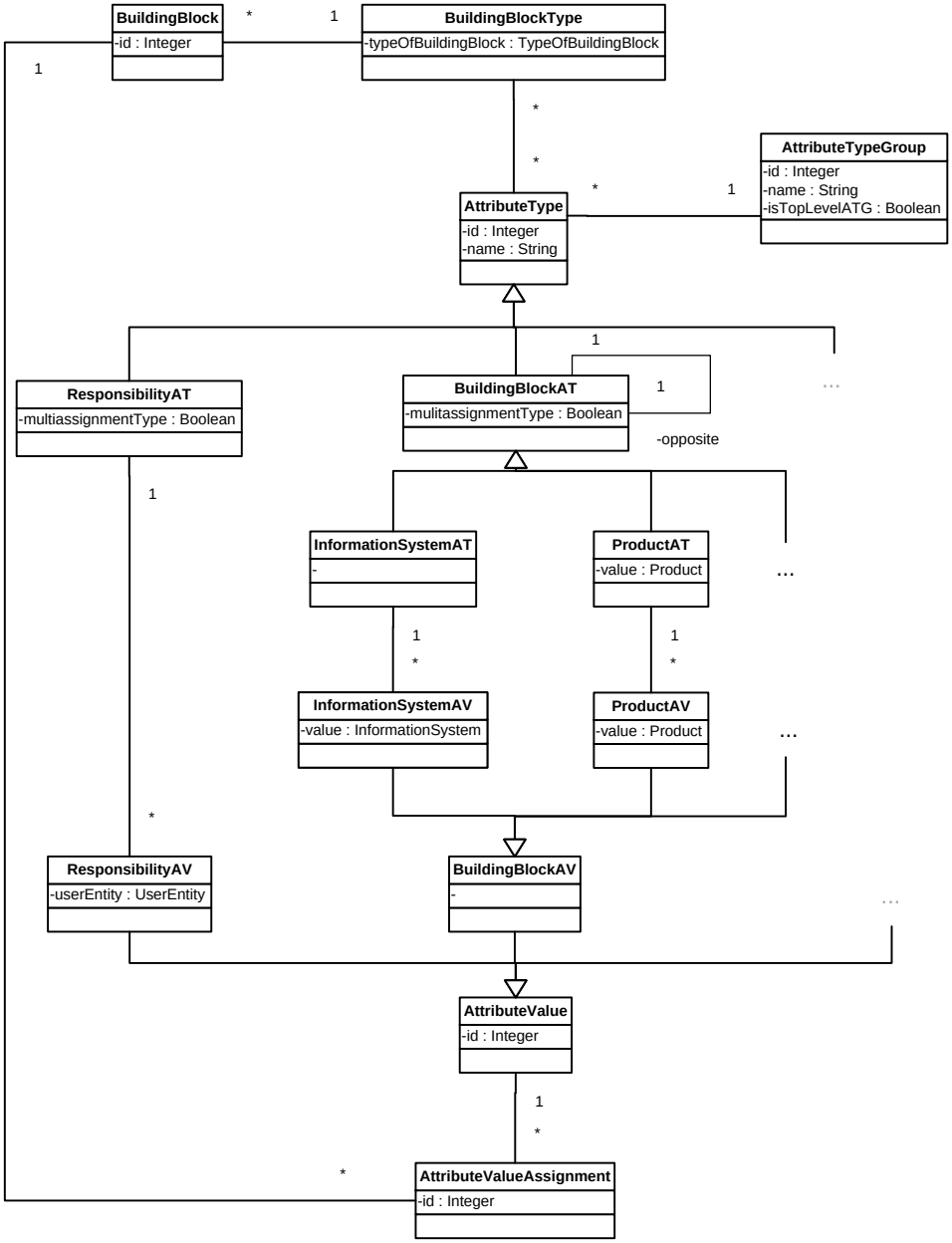

Abbildung 39 – Definition neuer Beziehungen durch Attributmechanismus

Abbildung 39 zeigt die beispielhafte Erweiterung um entsprechende *Merkmalstypen* für die *Buildingblocktypen Informationssystem* sowie *Produkt*. Eine derartige Erweiterung würde es für *Anwender* ermöglichen, dem von iteraplan bereitgestellten Informationsmodell, auf die gleiche Weise wie die Definition zusätzlicher *Verantwortlichkeitsmerkmale*, neue Beziehungen hinzuzufügen. Um dabei die Pflege der Beziehung an beiden Enden zu ermöglichen, wäre ein Ansatz ratsam, durch den bei der Definition einer zusätzlichen Beziehung, eine entsprechende Referenz in entgegengesetzter Richtung angelegt wird. Die beiden Beziehungen können dann über eine entsprechende Zuordnung gepaart werden (vgl. Autoassoziation „opposite“ der Klasse *BuildingBlockAT* in Abbildung 39). Dies würde bedeuten, dass die Pflege der selbst definierten Beziehung durch Hinzufügen bzw. Entfernen eines Wertes, automatisch durch zusätzliches Hinzufügen bzw. Entfernen in der entsprechend entgegengesetzten Relation erfolgt. Somit können auf einfache Weise zusätzliche Beziehungen des Informationsmodells modelliert, sowie durch automatische Konsistenzsicherungsmechanismen überprüft werden.

Um die Funktionalität dieses Mechanismus zu erweitern könnte die zusätzliche Typisierung bestimmter Beziehungen ermöglicht werden. So beschreibt [Bu10] beispielhaft die Differenzierung von Beziehungen des *Buildingblocktyps Projekt* (vgl. Abbildung 40).

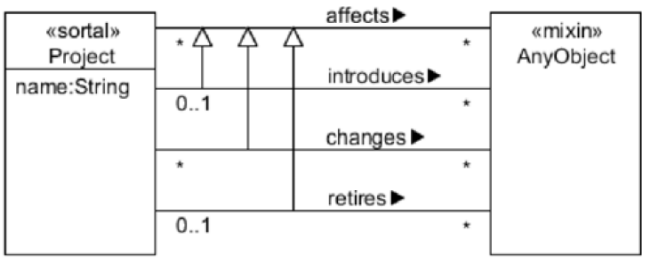

Abbildung 40 – Mögliche Beziehungstypen für Projekte (Quelle: [Bu10])

Denkbare Umsetzungen dieser Typisierung bestehen in der Implementierung weiterer Subklassen der jeweiligen *BuildingBlockAT*-Subklassen, oder in der Typisierung durch entsprechende Aufzählungstypen. Zusätzlich zu einer derartigen Erweiterung, kann auch die Konfiguration der vom Werkzeug unterstützten Visualisierungen (vgl. [it10]) angepasst werden. So könnten beispielsweise hierarchische Autoassoziationen, über eine passende Typisierung als solche gekennzeichnet werden, um so eine hierarchische Darstellung in Visualisierungen zu ermöglichen.

Die in Abschnitt 4.4.2.7 beschriebene Modellierung zeigt zusätzlich den Bedarf, neben den soeben beschriebenen Informationsmodellerweiterungen, ebenfalls die entsprechenden Informationsmodellreduzierungen durch die Entfernung von Autoassoziationen zu ermöglichen. Dies könnte auf vergleichsweise einfache Weise durch die Erweiterung der Rechteverwaltung erfolgen.

5.1.3 Klassen

Die in den Abschnitten 4.4.2.6 sowie 4.5.1 vorgestellten Modellierungen, beschreiben Möglichkeiten, *Buildingblocktypen* zu typisieren. Zusätzlich wird der *Buildingblocktyp Geschäftsprozess* durch die in Abschnitt 4.4.2.2 beschriebene Modellierung in „Geschäftsprozesse“ und „Services“ differenziert.

Allen in dieser Arbeit beschriebenen Typisierungen ist dabei gemeinsam, dass die Unterscheidung über *Diskriminatorattribute*, Namenskonventionen bzw. einer Kombination dieser beiden Möglichkeiten erfolgt. Dies zeigt den Bedarf zusätzlicher *Buildingblocktypen* und somit der Möglichkeit, dem iteraplan-Informationsmodell neue Klassen hinzuzufügen.

Um dies umzusetzen, und gleichzeitig den von iteraplan eingeschlagenen Weg, keine Informationsmodelländerungen im eigentlichen Sinn zu erlauben, könnte die Unterstützung eines „Typisierungsmechanismus“ auf Basis des bereits vorhandenen *Attributmechanismus* eingeführt werden. Dies würde die Definition sogenannter *Diskriminatorattribute* erfordern.

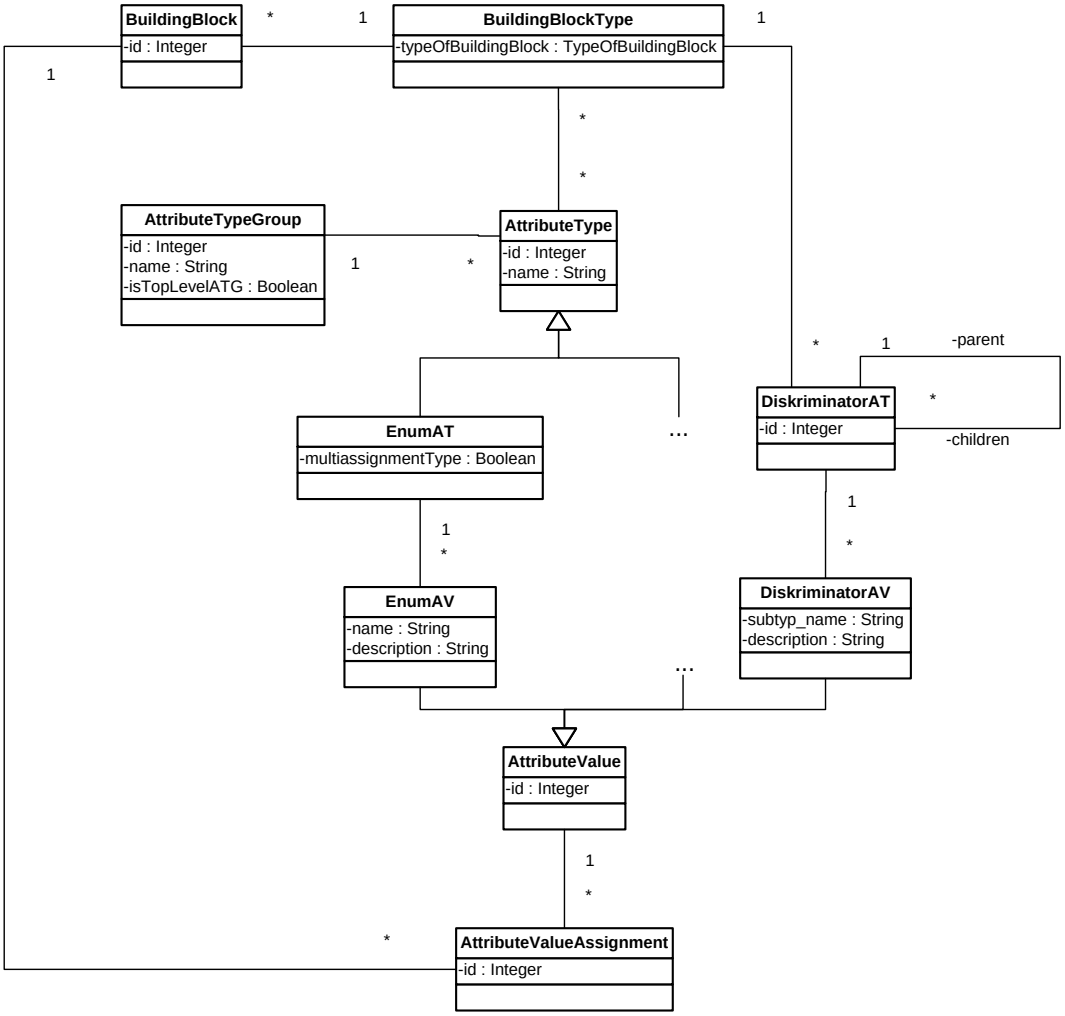

Abbildung 41 – Typisierung durch Attributmechanismus

Eine vergleichsweise einfache Umsetzung dieser Anforderung besteht in der Erweiterung des *Attributmechanismus* um die Klasse *DiskriminatorAT* („*Diskriminatormerkmal*“), welche größtenteils der Klasse *EnumAT* entspricht. Daneben erfordert eine solche Erweiterung auch die Definition einer entsprechenden *DiskriminatorAV* Klasse, zur Abbildung der möglichen Ausprägung (vgl. Abbildung 41). Eine hierarchische Anordnung dieser *Diskriminatormerkmale* durch die vorgeschlagene „parent-children“-Autoassoziation, kann dabei die Typisierung auf beliebig viele Ebenen erweitern.

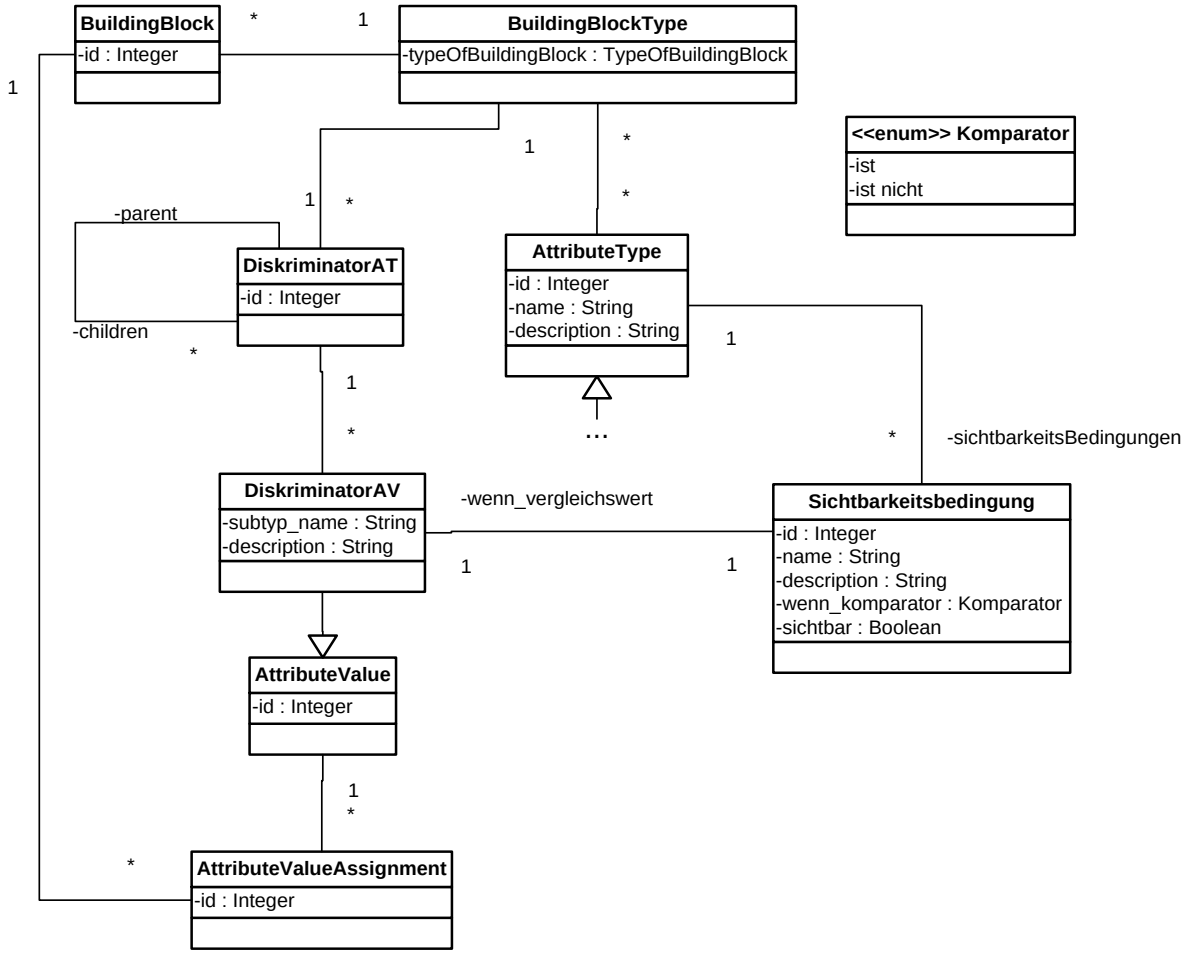

Abbildung 42 – Sichtbarkeitsbedingungen für Merkmalstypen

Neben diesen vergleichsweise geringen Anpassungen der iteraplan-Implementierung, würde diese Erweiterung des *Attributmechanismus* eine Reihe weiterer Anpassungen nach sich ziehen. So müsste zusätzlich ein ähnlich wie der in Abschnitt 5.1.1 beschriebener Bedingungsmechanismus implementiert werden, welcher es erlaubt den *Attributmechanismus* so anzupassen, dass *Merkmale* für einzelne *Subtypen* definiert werden können. D.h. *Merkmale* müssten unter den Werten entsprechender *Diskriminatorattribute* zugeordnet werden. Um diese Information abzubilden, könnten die bereits beschriebenen Ansätze (vgl. Abbildung 36 und Abbildung 41) kombiniert sowie um *Sichtbarkeitsbedingungen* erweitert werden. Abbildung 42 zeigt beispielhaft eine mögliche Umsetzung der Kombination dieser Ansätze. Mit Hilfe dieser

Erweiterung könnten so einzelne *Merkmalstypen*, unter Beachtung der jeweiligen Ausprägung des zugeordneten *Diskriminatorattributs*, ein- bzw. ausgeblendet werden. Dies würde somit eine dargestellte Typisierung der *Buildingblocktypen* auf der iteraplan-Benutzeroberfläche erlauben. Hierzu müssten die Änderungen selbstverständlich auch, unter vergleichsweise hohem Aufwand, konsistent auf allen Schichten der iteraplan-Implementierung (Datenhaltungs-, Geschäftslogik- sowie Präsentationsschicht) umgesetzt werden.

Abbildung 43 zeigt anhand eines UML-Objektdiagramms die beispielhafte Modellierung eines Ausschnitts der in Abbildung 15 beschriebenen Typisierung von *Informationssystemen* in „Standard"- und „Individualsoftware". Hierzu werden die drei *Merkmale* „lizenzkosten", „entwicklungszeit" und „gewartet" für den *Buildingblocktyp* der *Informationssysteme* definiert. Zur Typisierung des *Buildingblocktyps* werden zusätzlich das *Diskriminatorattribut* „Standard_Individual" sowie dessen mögliche Ausprägungen „Standard" und „Individual" angelegt.

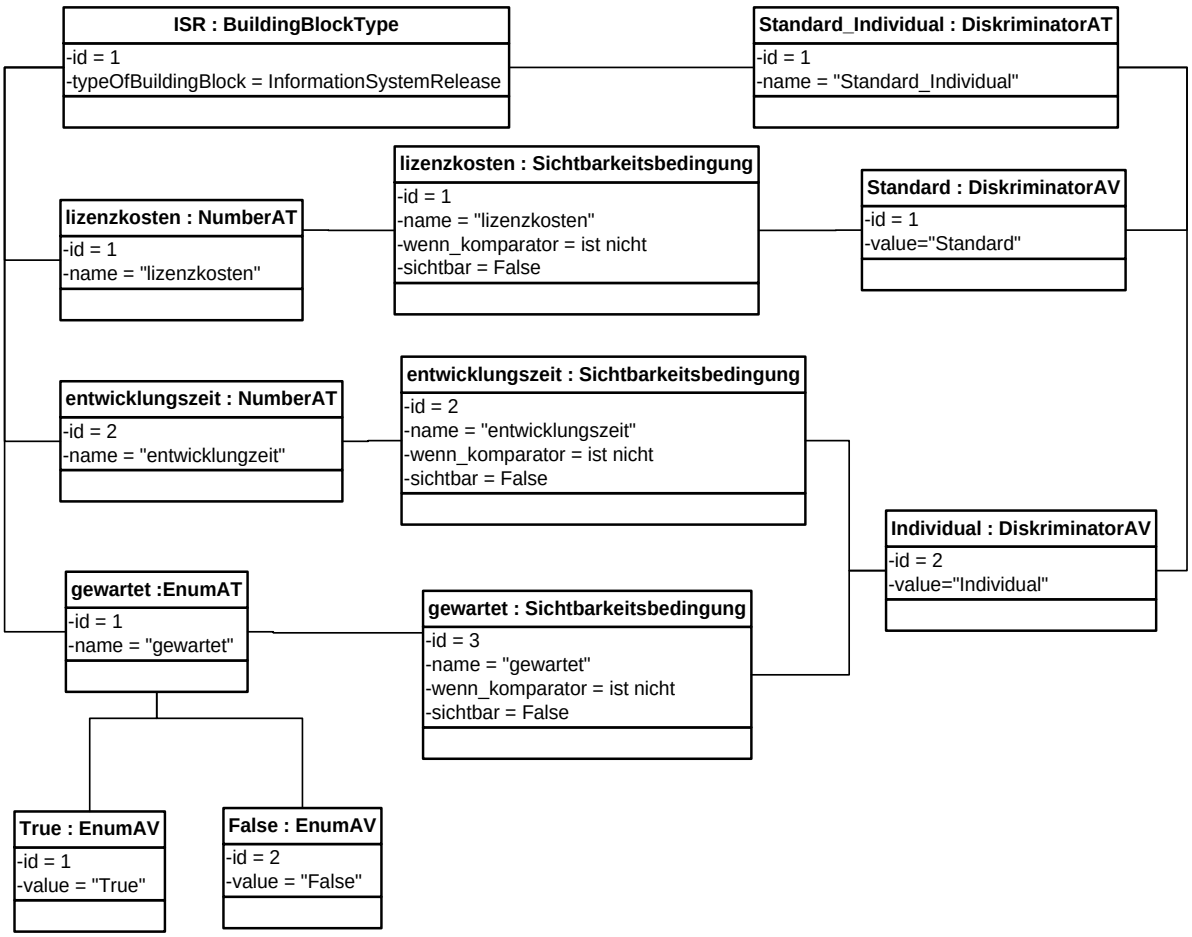

Abbildung 43 – Objektdiagramm für Sichtbarkeitsbedingungen

Zur Ausblendung der *Merkmale* entsprechend der in Abbildung 15 dargestellten Typisierung, können dann drei *Sichtbarkeitsbedingungen* angelegt werden. Wie Abbildung 43 zeigt, wird das *Merkmal* „lizenzkosten" immer dann ausgeblendet (vgl. *Sichtbarkeitsbedingung* „lizenzkosten"), wenn dem *Informationssystem* nicht der Wert „Standard" zugeordnet wurde.

Gleichermaßen werden entsprechende *Sichtbarkeitsbedingungen* für die verbleibenden *Merkmale* „entwicklungszeit“ und „gewartet“ gepflegt.

Die Unterscheidung, ob die *Sichtbarkeitsbedingung* dabei zur Sichtbarkeit oder zur Ausblendung des *Merkmals* führt wird durch das Boolean-Attribut „sichtbar“ indiziert. Das Beispiel setzt hier aus Übersichtlichkeitsgründen voraus, dass *Merkmale* grundsätzlich sichtbar sind, und explizit über entsprechende *Sichtbarkeitsbedingungen* ausgeblendet werden müssen. Ansonsten müssten zusätzlich korrespondierende *Sichtbarkeitsbedingungen* der Form „sichtbar = True“ modelliert werden, die den Komparator „ist“ verwenden.

Die beschriebene Erweiterung durch die Definition der zusätzlichen Klassen *DiskriminatorAT*, *DiskriminatorAV*, sowie *Sichtbarkeitsbedingung* ermöglicht es somit, durch den angepassten *Attributmechanismus Buildingblocktypen* weiter zu typisieren sowie die Pflege von *Merkmalen* die explizit für bestimmte Subtypen ein- bzw. ausgeblendet werden können.

5.2 Sonstige

Neben den im vorigen Abschnitt beschriebenen Anforderungen, welche die Anpassbarkeit des iteraplan-Informationsmodells adressieren, zeigen die in Kapitel 3.4 beschriebenen Modellierungen zusätzlich den Bedarf, iteraplan bzw. das zugrundeliegende iteratec-EA-Framework anzupassen.

5.2.1 Serviceorientierte Architekturen

Die sich bei mehreren untersuchten Organisationen wiederholende Modellierung einer SOA, zeigt die hohe Relevanz dieses Konzepts für das EAM. Auch wenn für das Konzept der *Serviceorientierten Architekturen*, genau wie für das EAM selbst gilt, dass sich bis heute keine allgemeine Definition für SOA durchsetzen konnte, gehören zu diesem Ansatz nach [Ma09] die drei Hauptbestandteile „Service Provider“, „Service Registry“ bzw. „Service Consumer“. Der *Service Provider* implementiert dabei eine Anwendung, welche von anderen Anwendungen über das Web aufgerufen werden kann (vgl. „Webservice“, [Ma09]). Dieser *Webservice* wird beim *Service Registry* registriert und somit veröffentlicht. Der *Service Consumer* kann das *Service Registry* nach *Webservices* durchsuchen, und so *Webservices*, sowie zur Nutzung dieser *Webservices* relevante Daten finden. Diese Daten betreffen dabei Informationen zur Nutzung des *Webservices*, wie die Adresse, an die ein entsprechender Aufruf zu richten ist, oder die Parameter, die ein solcher Aufruf enthalten muss. Durch die Verwendung dieser Daten kann der *Service Consumer* (Anwendung) die jeweiligen *Webservices* (Anwendung) aufrufen bzw. nutzen (vgl. Abbildung 44).

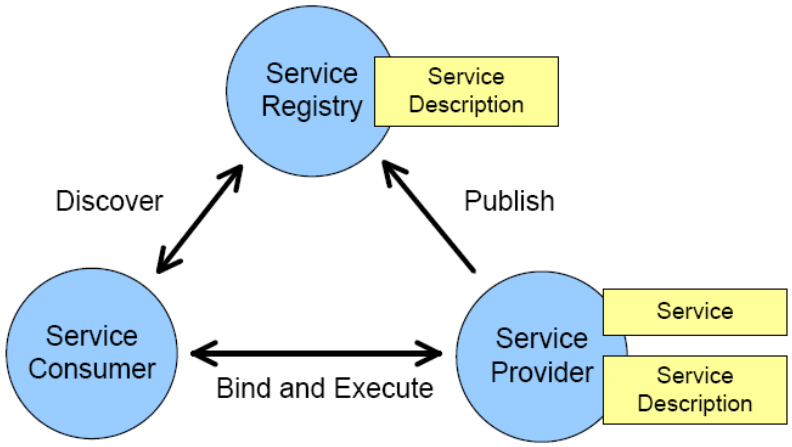

Abbildung 44 – Serviceorientierte Architekturen (Quelle: [Ma09])

Da nach dem von [Ha10] beschriebenen EA-Framework die IT-Unterstützung der *Geschäftsprozesse* eines Unternehmens im Vordergrund steht, werden für die SOA-Modellierung weder im Framework, noch in iteraplan explizite *Buildingblocktypen* für die in Abbildung 44 eingeführten Elementtypen beschrieben. Da somit in iteraplan zur Modellierung einer SOA ausschließlich die in Abschnitt 3.1 vorgestellten *Buildingblocktypen* zur Verfügung stehen, kann diese nur durch die Abstraktion der SOA-Konzepte (vgl. *Service Registry*, *Service Consumer*, *Service Provider* bzw. *Webservice*) abgebildet werden. Die Modellierung des *Service Registry* findet dabei keine Beachtung.

SOA	**iteraplan**
Service Consumer	*Informationssystem/Geschäftsprozess*
Service Provider	*Informationssystem/Schnittstelle*
Service Registry	
Webservice	*Informationssystem*

Tabelle 18 – Modellierung einer SOA in iteraplan

Abbildung 45 zeigt die beispielhafte Modellierung des SOA-Konzepts in iteraplan. Der *Webservice* wird als *Informationssystem* modelliert. Diesem *Webservice* werden über *Schnittstellen* die ebenfalls als *Informationssysteme* modellierten verwendeten „Subservices“ zugeordnet. Diese werden durch ihre Modellierung als „Teilinformationssystem“ einem logischen Informationssystem (vgl. Abbildung 45 „ISA“) zugeordnet. Dies bedingt die zusätzliche Notwendigkeit, über Namenskonventionen und bzw. oder entsprechende *Merkmale*, zwischen „fachlichen“ und „logischen“ *Informationssystemen* zu differenzieren.

Da diese Modellierung nicht durch automatische Konsistenzchecks überprüft bzw. geschützt werden kann, ist zu prüfen, ob die Einführung zusätzlicher *Buildingblocktypen* „Webservice“ bzw. „Service“, zu einer Verbesserung der Abbildungsmöglichkeiten von SOA beiträgt.

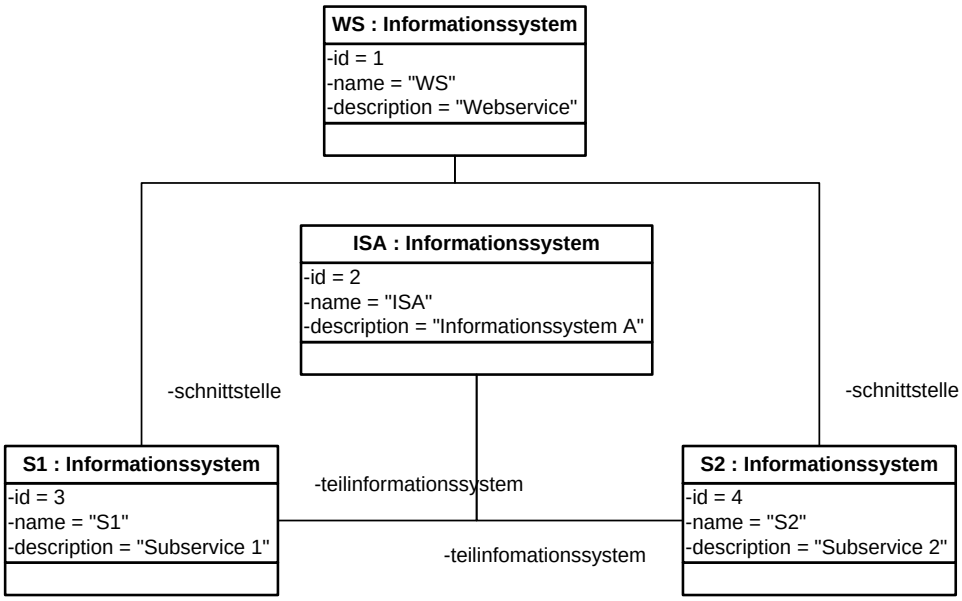

Abbildung 45 - SOA - Modellierung in iteraplan

5.2.2 Abfrage

Der größte Teil der in Abschnitt 4.4 untersuchten Modellierung konnte auf den Bedarf zusätzlicher Information zurückgeführt werden, der mit Hilfe der Definition eines bzw. mehrerer *Merkmale* abgedeckt werden kann. Jedoch wurden gewisse Informationen teilweise mehrfach abgebildet. So wurde beispielsweise die in Abbildung 29 dargestellte Unterscheidung von eigenen *Geschäftseinheiten* bzw. „Zulieferern“, sowohl durch eine Namenskonvention, als auch durch die hierarchische Anordnung der *Geschäftseinheiten* abgebildet. Zusätzlich wurde ein *Aufzählungsmerkmal* („BU_Type“) mit den Ausprägungen „Own“ bzw. „Supplier“ definiert, was ebenfalls zur Differenzierung dieser *Geschäftseinheiten* dient. Somit liegt die gewünschte Information dieser Unterscheidung im angepassten Informationsmodell der Organisation an drei Stellen vor. Die Konsistenz dieser Modellierung kann jedoch an keiner Stelle automatisch überprüft werden.

Wie Abbildung 29 zeigt, existieren auf der dritten Hierarchieebene korrespondierende *Geschäftseinheiten*. *Zulieferer* unterscheiden sich dabei von eigenen *Geschäftseinheiten* lediglich über das Präfix „Supplier“. Diese Namenskonvention liegt im iteraplan-Informationsmodell begründet, welches keine verschiedenen *Geschäftseinheiten* mit gleichen Namen zulässt. An dieser Stelle wäre jedoch auch eine optionale Ausweitung dieser Einschränkung auf den „hierarchischen Namen“ sinnvoll.

Die hierarchische Anordnung der *Geschäftseinheiten* dient zur Darstellung der Organisationsstruktur, welche sich so in den von iteraplan bereitgestellten grafischen Auswertungen darstel-

len lässt. Der Zweck des *Aufzählungsmerkmals* „BU_Type“ besteht in der zusätzlichen Strukturierung grafischer Auswertungen. iteraplan bietet bei der Konfiguration grafischer Auswertungen die Möglichkeit, sowohl die Farbe als auch die Größe der in der Grafik enthaltenen Elemente, in Abhängigkeit der jeweiligen Ausprägung selbst definierter *Aufzählungs*- bzw. *Zahlenmerkmale* darzustellen. Die konsistente Modellierung derartiger Daten kann über *Bedingungen* (vgl. Abschnitt 5.1.1) ermöglicht werden. Hierzu ist dieses Konzept jedoch wie bereits beschrieben, auf die Formulierung von *Bedingungen* zu erweitern, die eine Prüfung von Attributen und *Merkmalen* referenzierter Objekte erlauben.

Für die Auswahl der in einer Grafik enthaltenen Instanzen eines *Buildingblocktyps*, stellt iteraplan einen „Abfragemechanismus“ zur Verfügung. Dieser erlaubt die Auswahl der Elemente durch die Formulierung von Abfragen bezüglich aller Standardattribute sowie *Merkmale* (vgl. [it10]) des *Buildingblocktyps* sowie von diesem Typ direkt referenzierte Objekte. Hierzu können, je nach Typ des Attributs bzw. *Merkmals*, die in Tabelle 17 beschriebenen Komparatoren verwendet werden. Darüber hinaus können verschiedene solcher Abfragen über die boolschen Operatoren „und“ bzw. „oder“ miteinander verknüpft werden. Eine Formulierung einer Abfrage, welche Attribute oder *Merkmalstypen* referenzierter Objekte über mehrere Beziehungen hinweg adressiert, ist dagegen nicht möglich. Somit besteht ein weiterer Grund für die Einführung der beschriebenen Namenskonvention bzw. des *Aufzählungsmerkmals* „BU_Type“ zur Unterscheidung der *Geschäftseinheiten* durch den Abfragemechanismus. Um das bereits beschriebene Beispiel aus Abbildung 29 fortzuführen, kann somit der fiktive Anwendungsfall einer Auswahl aller *Geschäftseinheiten*, deren Vorfahren die *Geschäftseinheit* „Suppliers“ enthalten, nicht direkt unterstützt werden. Erst die Überprüfung des Attributs „Name“ bzw. des *Merkmalstyps* „BU_Type“ erlaubt dies.

Das beschriebene Beispiel sowie die in Abschnitt 4.4.2.7 untersuchte Modellierung zeigen jedoch den Bedarf der Abfrage dieser Information auf. Somit ist auch hier zu prüfen, ob eine transitive Abfrage über mehrere Beziehungen des Informationsmodells oder über Autoassoziationen eines *Buildingblocktyps* unterstützt werden sollte.

5.2.3 Mixins

Die Untersuchungen in Kapitel 4 haben mehrfach gezeigt, dass von iteraplan implementierte Beziehungen verschiedener *Buildingblocktypen,* von einigen Organisationen nicht genutzt werden. Ein häufig betrachtetes Beispiel hierfür, ist die hierarchische Anordnung einzelner *Buildingblocktypen*. Wie in Abschnitt 4.2.2 beschrieben, bietet iteraplan für diesen Sachverhalt die Möglichkeit, die entsprechende Beziehung über die Rechteverwaltung aus der iteraplan-Oberfläche zu entfernen.

An dieser Stelle wäre jedoch auch ein Ansatz denkbar, welcher durch die in Abbildung 6 beschriebenen Modellierungsmuster indiziert wird. Zur Steigerung der Flexibilität des Werkzeugs könnten sog. *Mixins* definiert werden. *Mixins* entsprechen dabei nach [Gu04], ähnlich wie abstrakte Superklassen, Stereotypen die bestimmte Eigenschaften sowie damit verbundene Funktionalität kapseln. Abbildung 46 zeigt beispielhaft eine vereinfachte Modellierung der *Mixins* für die bereits beschriebenen iteraplan-Modellierungsmuster (vgl. Abbildung 6).

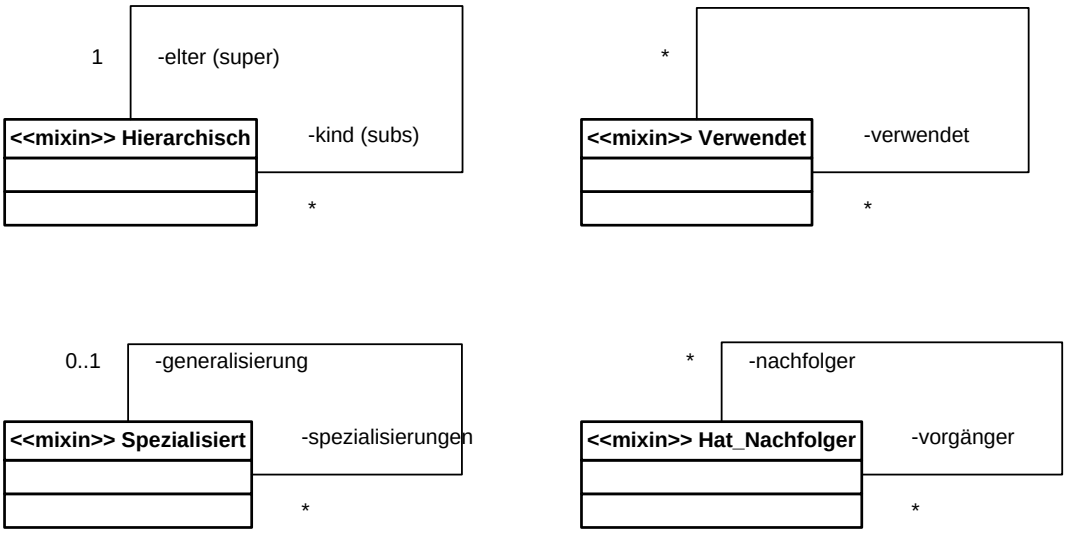

Abbildung 46 – Mixins der iteraplan Modellierungsmuster

Dieser *Mixin*-Ansatz könnte zur Steigerung der Flexibilität des Werkzeugs beitragen, indem ungenutzte Aspekte des iteraplan-Informationsmodells auf ebenfalls einfache Weise entfernt, und gleichzeitig weitere Elemente des Informationsmodells flexibel an die jeweiligen Anforderungen des Unternehmens angepasst werden können. So könnte beispielsweise die Rechteverwaltung durch eine mögliche Modellierung einer „nachfolger-vorgänger"-Beziehung von Benutzern erleichtert, oder die Modellierung von Abhängigkeiten verschiedener Attribute über das *Mixin* „Verwendet" ermöglicht werden.

Diese Erweiterung des Werkzeugs, bedingt jedoch den zusätzlichen Bedarf eines Konfigurationsmechanismus, der es *Anwendern* auf einfache Weise ermöglicht, *Mixins* für Elemente des Informationsmodells zu aktivieren bzw. zu deaktivieren. Zusätzlich könnte ein solcher Konfi-

gurationsmechanismus in einem nächsten Schritt so erweitert werden, dass die Definition eigener *Mixins* unterstützt wird.

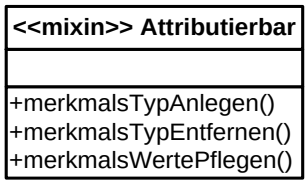

Abbildung 47 – Mixin „Attributierbar“

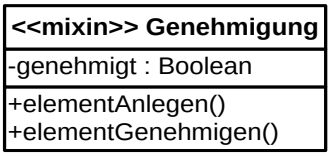

Abbildung 48 – Mixin „Genehmigung“

Neben den vergleichsweise einfachen *Mixins* (vgl. Abbildung 46), wäre auch eine Restrukturierung des in Abschnitt 3.3 beschriebenen *Attributmechanismus* durch ein *Mixin* „Attributierbar“ denkbar (vgl. Abbildung 47). Sinnvoll wäre an dieser Stelle auch die Definition von entsprechenden *Mixins* für alle *Merkmalstypen*, um eine noch flexiblere Anpassung des Werkzeugs zu ermöglichen.

Somit könnten Unternehmen entscheiden, welche Elemente des Informationsmodells durch den *Attributmechanismus* erweiterbar sein sollen. Dadurch könnten neben *Benutzern*, auch *Merkmalstypen* selbst um weitere *Merkmale* erweitert werden, was die Ausdruckskraft der vom Werkzeug unterstützten Modellierungsmöglichkeiten erhöhen könnte. Denkbare Anwendungsfälle wären beispielsweise *Verantwortlichkeitsmerkmale* für andere *Merkmalstypen*, sowie eine aus Entwicklungssicht vereinfachte Erweiterung des *Attributmechanismus* um weitere *Merkmalstypen*. Ein anderer Anwendungsfall, der durch die Verwendung von *Mixins* abgedeckt werden könnte besteht in der *orthogonalen Typisierung* von *Buildingblocktypen*. Abbildung 48 zeigt die beispielhafte Modellierung des *Mixins* „Genehmigung“, welche die in Abschnitt 4.5.1 beschriebene Typisierung von *Projekten*, darstellen kann.

Daneben können auch andere Anforderungen über die Implementierung von *Mixins* umgesetzt werden. Abbildung 40 indiziert hierzu bereits einen entsprechenden Ansatz für Beziehungen.

Die Erweiterung des Werkzeugs, welche die beschriebene Funktionalität ermöglicht, könnte so den größten Teil der in diesem Kapitel beschriebenen Anforderungen umsetzten, doch würde dieser auch den größten Anpassungsaufwand mit sich ziehen. Daher ist zu prüfen, ob der *Mixin*-Ansatz für die künftige Entwicklung des Werkzeugs in Betracht gezogen werden kann, oder ob dieser dem *methodenbasierten* Ansatz (vgl. Abschnitt 2.2) des Werkzeugs widerspricht.

6 Zusammenfassung und Ausblick

Um die Arbeit abzuschließen fasst dieses letzte Kapitel die wichtigsten Erkenntnisse, die durch die Erhebung zusätzlicher Anforderungen gewonnen wurden noch einmal zusammen. Daneben werden die in Kapitel 5 beschriebenen Erweiterungsvorschläge bewertet und mögliche Ansätze für die zukünftige Entwicklung des untersuchten Werkzeugs formuliert.

6.1 Zusammenfassung

Das Ziel dieser Arbeit bestand in der Untersuchung der Modellierungsmöglichkeiten des EAM-Werkzeugs iteraplan, das einen nach der in Abschnitt 2.2 beschriebenen Kategorisierung *methodenbasierten* Ansatz verfolgt. Nach einer Einführung in den Kontext des EAMs wurde das untersuchte Werkzeug sowie dessen zugrundeliegendes EA-Framework detailliert vorgestellt.

In einem nächsten Schritt wurden die allgemein möglichen Informationsmodellanpassungen sowie deren Unterstützung durch das Werkzeug beschrieben. Dies ermöglichte die Analyse der Übertragbarkeit von verschiedenen EA-Informationsmodellen einzelner Unternehmen auf das Informationsmodell des Werkzeugs. Zusätzlich wurden anonymisierte iteraplan-Datensätze sowie die darin enthaltenen angepassten EA-Informationsmodelle der jeweiligen Unternehmen untersucht (vgl. Kapitel 4).

Als Ergebnis dieser Analysen konnte der Bedarf zusätzlicher Adaptionsmechanismen identifiziert werden, die zum Untersuchungszeitpunkt nicht vom Werkzeug unterstützt wurden. Dieser Bedarf an zusätzlichen Anpassungsmöglichkeiten wurde in Kapitel 5 detailliert beschrieben. So wurden u.a. die Notwendigkeit der Modellierung abhängiger Attribute sowie die Erweiterung des Informationsmodells um Beziehungen und Klassen aufgezeigt.

Neben der Erhebung von zusätzlichen Anforderungen wurden mögliche Lösungen vorgeschlagen, die das Werkzeug um die entsprechenden Adaptionsmechanismen erweitern. In Abschnitt 5.2.3 wurde schließlich ein weiterer Ansatz angeführt, dessen Umsetzung gleichzeitig die zuvor beschriebenen Erweiterungen abdecken kann.

6.2 Ausblick

Nachdem die Untersuchungen der Anpassungsmöglichkeiten von iteraplan den Bedarf zusätzlicher Adaptionsmechanismen gezeigt haben, ist nun zu prüfen, inwieweit die vorgeschlagenen Lösungen bei der künftigen Entwicklung des Werkzeugs berücksichtigt werden. Kapitel 5 stellt hierzu eine Sammlung an möglichen Erweiterungsansätzen zur Verfügung. Diese können einzeln oder in kombinierter Form verfolgt bzw. umgesetzt werden. Die in Abschnitt 5.2.3 eingeführte Erweiterung hebt sich dabei von den verbleibenden Ansätzen ab, da dieser gleichzeitig alle zuvor beschriebenen Anforderungen abdecken kann. Dieser *Mixin*-Ansatz bedeutet demnach eine umfangreiche Ausweitung der Anpassungsmöglichkeiten von iteraplan. Die Informationsmodellanpassungen die hierdurch ermöglicht werden steigern dabei die Aussagekraft der Modellierungsmöglichkeiten, die bisher auf *modellorientierte* Werkzeuge beschränkt ist. Daher ist zu prüfen, ob diese Erweiterung mit dem *methodenbasierten*-Ansatz von [Ha10] zu vereinbaren ist.

Zusätzlich ist zu beachten, dass die Umsetzung eines bzw. mehrerer der in Kapitel 5 vorgeschlagenen Ansätzen immer eine detaillierte Aufwands- und Umsetzbarkeitsanalyse voraussetzt. Neben den Änderungen in der *Geschäftslogik*- und der *Präsentationsschicht* betrifft dies im Speziellen die Auswirkungen auf die von iteraplan unterstützten Visualisierungen. So ist beispielsweise zu erwarten, dass der Bereich der *grafischen Auswertungen* bei einer Implementierung der in den Abschnitten 5.1.2 und 5.1.3 beschriebenen Erweiterungen umfangreich angepasst werden muss. Auch hier ist zu prüfen, inwieweit die Erweiterungen mit den derzeit unterstützten Visualisierungen vereinbart werden können. Diese Analysen müssen jedoch im Rahmen der Weiterentwicklung des Werkzeugs durchgeführt werden, da diese Arbeit ausschließlich abstrakte Lösungsansätze für die zusätzlich erhobenen Modellierungsanforderungen beschreibt.

Abschließend bleibt festzuhalten, dass das untersuchte Werkzeug für Unternehmen die Möglichkeit bietet, EAM auf vergleichsweise einfache und schnelle Weise einzuführen bzw. zu betreiben. Das implementierte Informationsmodell bildet hierzu eine Basis, die an die unternehmensspezifischen Anforderungen angepasst bzw. entsprechend erweitert werden kann. Zur zusätzlichen Steigerung der Anpassungsmöglichkeiten können die in Kapitel 5 beschriebenen Lösungsansätze weiter geprüft und umgesetzt werden.

7 Literaturverzeichnis

[Bu08] *Buckl, S.; Ernst, A. M.; Lankes, J.*: *Prof. Dr. Matthes, F.:* EAM Pattern Catalog 1.0. TU München, 2008.

[Bu09] *Buckl. S; Franke, U.; Holschke, O.*; *Matthes, F.; Schweda, C.; Sommerstad, T.; Ullberg, J.*: A Pattern-based Approach to Quantitative Enterprise Architecture Analysis. 15th Americas Conference on Information Systems (AMCIS), San Francisco, USA, 2009.

[Bu10] *Buckl, S.; Matthes, F.; Schweda, C.*: Conceptual Models for Cross-cutting Aspects in Enterprise Architecture Modeling. Joint 5th International Workshop on Vocabularies, Ontologies, and Rules for the Enterprise (VORTE 2010), Vittoria, Brazil, 2010.

[De06] *Delfmann, P.*: Adaptive Referenzmodellierung. Westfälische Wilhelms-Universität Münster, 2006.

[Do04a] *Department of Defence Architecture Framework Working Group*: DoD Architecture Framework Version 1.0, Volume I: Definitions and Guidelines. USA 2004.

[Do04b] *Department of Defence Architecture Framework Working Group*: DoD Architecture Framework Version 1.0, Volume II: Product Descriptions. USA 2004.

[En08] *Engels, G.; Hess, A.; Humm, B.; Juwig, O.; Lohmann, M.; Richter, J.-P.*: Quasar Enterprise: Anwendungslandschaften serviceorientiert gestalten. 1. Auflage. Dpunkt, Heidelberg 2008.

[Fi07] *Fischer, R.; Aier, S.; Winter, R.*: A Ferderated Approach to Enterprise Architecture Model Maintenance. University of St. Gallen, 2007.

[Gu04] *Guizzardi, G; Wagner, G.; van Sinderen, M.*: A Formal Theory of Conceptual Modeling Universals. Procedings of the Workshop on Philosophy and Informatics (WSPI). Köln 2004

[Ha10] *Hanschke, I.*: Strategisches Management der IT-Landschaft. 2. Auflage. Hanser, München 2010.

[Hi10] *Hirschfeld, R.; Keller, W.*: Architekturframeworks und Blueprints. Hasso Plattner Institut, 2010.

[it10] *iteratec GmbH*: www.iteraplan.de, aufgerufen am 01.09.2010

[Kr10] *Krcmar, H.; Schwarzer, B.*: Wirtschaftsinformatik: Grundlagen betrieblicher Informationssysteme. Schäffer-Poeschel, Stuttgart, 2010.

[Ma08] *Matthes, F.; Buckl, S.; Leitl, J.; Schweda, C. M.*: Enterprise Architecture Management Tool Survey 2008. TU München, 2008.

[Ma09] *Matthes, F.; Büchner, T.; Steinhoff, A.*: Software Architectures : 6. Architecture for Systems of Systems. TU München, 2009

[Mal10] *Malik, N.*: A reasonable canonical definition of Enterprise Architecture. http://blogs.msdn.com/b/nickmalik/archive/2010/08/08/a-reasonable-canonical-definition-of-enterprise-architecture.aspx, aufgerufen am 27.08.2010

[Pe07a] *Peyret, H.*, with *Carini, A.; Hoekendijk, C.; King, O.; Leganza, G.; McCormack, M.*: The Forrester Wave™: Enterprise Architecture Tools, Q2 2007. Forrester Research, 2007.

[Pe07b] *Peyret, H.*, with *Hoekendijk, C.; Leganza, G.*: Best Practices For Choosing EA Tools. Forrester Research, 2007.

[Pe09a] *Peyret, H.*, with *An, M.; Leganza, G.; Smillie, K.*: The Forrester Wave™: Business Process Analysis, EA Tools And IT Planning, Q1 2009. Forrester Research, 2009.

[Pe09b] *Peyret, H.*, with *An, M.; Cullen, A.*: Open Source Solutions For EA Tool Needs Are Progressing. Forrester Research, 2009.

[Pr09] *Pragmatic EA Ltd*: The 160 Character Challenge. www.pragmaticea.com/160challenge.asp, aufgerufen am 20.08.2010

[Sk04] *Schekkerman, J.*: How to survive in the Jungle of Enterprise Architecture Frameworks. 2. Auflage. Trafford Publishing, Canada 2004.

[So10] SourceForge: http://sourceforge.net/projects/iteraplan/files/, zuletzt aufgerufen am 24.08.2010

[Su09] *Sull, D.*: The Upside Of Turbulence. HarperCollins Publishers. New York, 2009

[To09] *The Open Group*: TOGAF™ Version 9. 1. Auflage. Van Haren Publishing, 2009.

[Za87] *Zachman, J.*: Framework for Information Systems Architecture. In: IBM Systems Journal, Vol. 26, No. 3, 1987.