Richard Lenz, Ullrich Hasenkamp,
Wilhelm Hasselbring, Manfred Reichert (Hrsg.)

EAI 2005
Enterprise Application Integration

Tagungsband des 2. GI-/GMDS-Workshops
zum Thema Enterprise Application Integration

Philipps-Universität Marburg, 30.06. – 01.07.2005

Vorwort

Die heutige Flut von Informationsquellen im betrieblichen Umfeld bestimmt die immer weiter zunehmende Notwendigkeit der Integration von Anwendungssystemen und Datenbanken innerhalb der betrieblichen Datenverarbeitung und über Unternehmensgrenzen hinweg. Enterprise Application Integration (EAI) bezeichnet in diesem Kontext die Planung, die Methoden und die Software, um heterogene, autonome Anwendungssysteme unternehmensweit oder -übergreifend zu integrieren.

Der EAI-Workshop findet zum zweiten Mal nach der erfolgreichen Auftaktveranstaltung am OFFIS-Institut in Oldenburg im vergangenen Jahr statt. Der Workshop EAI'05 ist eine gemeinsame Veranstaltung der GI-Arbeitskreise „Enterprise Architecture" und „Software Architekturen", der GMDS-/GI-Arbeitsgruppe „KIS - Informationssysteme im Gesundheitswesen" sowie des Klinikums der Philipps-Universität Marburg. Das Thema EAI soll auf dem Workshop aus übergreifender, ganzheitlicher Sicht betrachtet werden. Sowohl die grundlegenden Problemstellungen, wie semantische Integration und Wissensmanagement in verteilten heterogenen Systemen, als auch pragmatische Lösungsansätze und Vorgehensmodelle auf der Basis von Integrationswerkzeugen und innovativen IT-Infrastrukturen stehen im Fokus.

Berichte aus spezifischen Anwendungsbereichen, wie Bankwesen, Bauwirtschaft und Gesundheitswesen zeigen Anforderungsprofile und konkrete Lösungen auf. Einen Schwerpunkt bildet die integrierte Gesundheitsversorgung mit Beiträgen zur Anwendungsintegration im Krankenhaus sowie zu verschiedenen Fragestellungen im Zusammenhang mit der Integration in Gesundheitsnetzen. Darüber hinaus werden mit verschiedenen Beiträgen zur prozessorientierten Integration aktuelle Bestrebungen thematisiert, die in verschiedenen Anwendungsdomänen von Interesse sind.

Zum Gelingen eines Workshops tragen viele bei. Wir möchten uns bei allen Vortragenden, insbesondere bei den eingeladenen Rednern, und ihren Koautoren bedanken. Ebenso herzlich danken wir den Mitgliedern des Programmkomitees sowie allen externen Gutachtern für ihre Sorgfalt und Mühe. Unser besonderer Dank gilt den Mitarbeitern des Instituts für Medizinische Informatik und des Instituts für Wirtschaftsinformatik für die lokale Organisation. Danken möchten wir auch dem Klinikum der Philipps-Universität Marburg für die Bereitstellung der Räumlichkeiten.

Wir begrüßen Sie, liebe Workshopteilnehmer, herzlich zur EAI'2005 in Marburg und hoffen, dass Sie das Workshop-Programm und die Abendveranstaltung abwechslungs- und erkenntnisreich finden werden.

Richard Lenz, Ullrich Hasenkamp, Wilhelm Hasselbring, Manfred Reichert

Veranstalter:

GI-EMISA-Arbeitskreis „Enterprise Architecture“
GI-OOSE-Arbeitskreis „Software Architekturen“
GMDS/GI-Arbeitsgruppe KIS
Klinikum der Philipps-Universität Marburg

Workshop-Leitung:

Wilhelm Hasselbring (Universität Oldenburg)
Richard Lenz (Universität Marburg)
Manfred Reichert (Universität Twente, Niederlande)

Lokale Organisation

Ullrich Hasenkamp (Universität Marburg)
Richard Lenz (Universität Marburg)

Programmkomitee

Stefan Conrad (Universität Düsseldorf)
Holger Günzel (IBM Unternehmensber. GmbH)
Ullrich Hasenkamp (Universität Marburg)
Wilhelm Hasselbring (Universität Oldenburg)
Stefan Jablonski (Universität Erlangen)
Arne Koschel (Deutsche Post AG)
Klaus Kuhn (TU München)
Richard Lenz (Universität Marburg)
Jan Mendling (Wirtschaftsuniversität Wien)
Manfred Reichert (Universität Twente)
Stefanie Rinderle (Universität Ulm)
Jörg Ritter (BTC AG)
Günter Steyer (SeeBeyond)
Robert Winter (Universität St. Gallen)

Inhaltsverzeichnis

Eingeladene Beiträge

Wissenschaftliches Programm

Anwendungsintegration im Krankenhaus

Gesundheitsnetze

Prozessorientierte Integration

Integrationsanforderungen auf verschiedenen Ebenen

Eingeladene Beiträge

Ontologies in the Software Engineering process

Wolfgang Hesse

Fachbereich Mathematik und Informatik, Univ. Marburg,
Hans Meerwein-Str., D-35032 Marburg
hesse@informatik.uni-marburg.de

Abstract: The term *ontology* has become popular in several fields of Informatics like Artificial Intelligence, Agent systems, Database or Web Technology. Deviating from its original philosophical meaning, in the context of Computer Sciences the term *ontology* stands for *a formal explicit specification of a shared conceptualization*. Software Engineering (SE) is a field where conceptualisation plays a major role, e.g. in the early phases of software development, in the definition, use and re-use of software components and as a basis for their integration. Thus, ontologies are likely to invade the SE field as well soon.

In this contribution, conceptual modeling as practiced in SE and Information Systems projects is contrasted with the ontology approach. The corresponding life cycle models for their development are compared. Finally, some perspectives of an *Ontology-based Software Engineering (OBSE)* approach are outlined.

1 Introduction: What is an ontology?

In several fields of Informatics such as Artificial Intelligence, Agent systems, Information systems, Database or Web technology, the term "ontology" has widely been used during the last years. Various languages and techniques for their definition and use have been published and disseminated. Recently, the ontology approach is being considered and debated in parts of the Software Engineering (SE) and the Business Applications communities – mainly for its proximity to the (conceptual and business) modelling fields, to the UML language and the fields of software component re-use and integration.

Before we start to introduce an old term like *ontology* into a new branch of Computer Science it is worthwhile to reflect on its traditional use in the area of philosophy where it is originating from. Originally *ontology* means the "science of being" - more accurately: it deals with the *possibilities* and *conditions* of the being. In other words, philosophers ask what it means "to be", what are the prerequisites and limitations of "being", its origins and future horizons. Such questions lead immediately into further questions on what we might *know about* the being of things and facts and thus ontology is strongly related to *epistemology* and *cognition theory*.

In contrast to most philosophers, computer scientists adopts a naïve, "realistic" approach. Normally, we accept without further questions as "being" what we can

observe, feel, touch or derive form these sensations by reasoning and other mental reflection. However, views and convictions about the same "being object" might deviate from one observer to the next. This makes communication and coming to an agreement often a difficult or even unsolvable task. Our observation of and communication on "being things" is hampered by at least two principal reasons (cf. fig 1):

(1) We cannot be sure that what we observe really "is" or "exists" in the fundamental ontological sense.

(2) We communicate our knowledge on what "is" or "exists" using particular languages (including the so-called "natural" ones) which always are limited in their expressiveness and force us to reduction, incompleteness and imprecision.

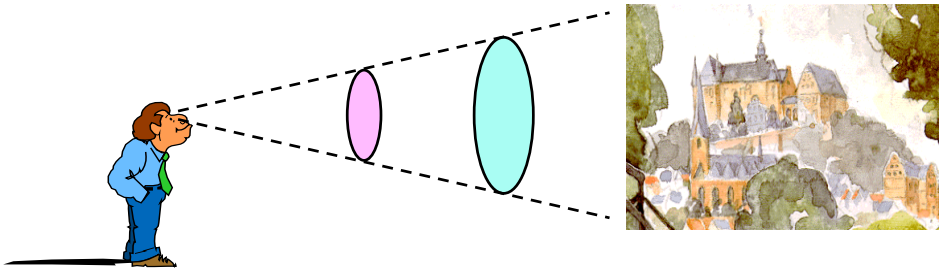

Fig. 1: Filtered view on the "being"

Therefore an ontology cannot express *"what is"* but at most what *we believe to know* about some part of the world and what we are able to *communicate* through the language at our disposal. In Informatics, such a communication takes place at several points: between users and developers of a planned or existing software system while analysing their requirements and (re-) designing or implementing the system functionality, between users and systems or their components while working through a user interface or between components of systems or automated devices while co-operating with each other to solve a given task.

In Artificial Intelligence, the notion *ontology* was introduced in the 1970ies - mainly to denote representations of particular areas of knowledge as needed e. g. for automated devices (robots, agents) for their orientation while performing given tasks. Later, this notion was slightly generalized leading (for example) to the following definitions:

> *"Ontologies provide a shared and common understanding of a domain that can be communicated between people and application systems."* (Hensel, [12])
>
> *"The role of ontologies is to capture domain knowledge in a generic way and to provide a commonly agreed upon understanding of a domain. The common vocabulary of an ontology, defining the meaning of terms and their relations, is usually organised in a taxonomy and contains modelling primitives such as classes, relations, functions, and axioms."* (Mädche et. al, [20])

Probably the most condensed definition originates from T. Gruber ([9]):

> *"An ontology is a formal explicit specification of a shared conceptualization"*

In this context, the use of "ontologies" (in plural form) does indeed make sense: At first, there are many parts of the world which can be conceptualised (while one single, universal conceptualisation of the "whole world" is beyond any limits of reasoning and data storage capabilities). Secondly, for any given knowledge domain infinitely many possible conceptualisations might be formed. Thus for most domains several competing ontologies co-exist in practice.

The use of the notion "ontology" in Informatics is rather controversial – some philosophers even consider it a misuse [17]. Indeed, "*standardised reference model for a certain application domain"* would be more exact and self-explaining – but would it be more practical? At least, there seems to be some congruence between Gruber's definition and the constructivist position in modern philosophy: We cannot know what the world is like but we can construct (and communicate) world views which are to be assessed rather through criteria of *practicability* than by those of *truth*. If we consider software development as a part of "reality construction" [6], ontologies (in this restricted sense) might find their adequate place in it.

2 Ontologies in and for Software Engineering

Which roles might ontologies play in the Software Engineering field? Basically, we see two close connections:

- *Ontologies in the Software Engineering process:* Modularisation, distribution, reuse and integration of software components and systems is among the central SE issues from its first days. The more these tasks are extended and automated, the more important gets the definition and use of ontologies as conceptual basis of such components. Recent software development methodologies such as model-driven development (MDD) [19] favour models as the central knowledge base from which different implementations may be derived. If systems or components are to exchange knowledge, this will happen more on the model than on the implementation level. In the era of coalescing application landscapes a big potential for defining, implementing, disseminating and using ontologies is evolving.
- *Ontologies for the Software Engineering domain:* SE is an own scientific and professional domain with an own structure and terminology. Since it is a rather new domain the task of formulating one (or several?) ontology/ies for this field is a necessary and challenging task. We mention two approaches: the *SWEBoK project* in the English-spoken community [3] and the German *terminology network* (*"Begriffsnetz"*, [8]).

In the rest of this article I will focus on the first point: *Ontologies in the Software Engineering process***.**

3 Ontologies: just a new label for old conceptual models?

At least since the days when P. Chen invented his Entity/Relationship (E/R-) model and published the corresponding diagrams, *Conceptual Modelling (CM)* is considered a key activity for database development. In the SE field, the importance of data

modelling was recognised and the E/R techniques became popular a few years later. Since then they have strongly influenced most SE analysis and design methodologies, languages and tools. E/R diagrams were diligently modified, re-defined and extended (e.g. by inheritance mechanisms) and they became the central modelling tool for almost every SE analysis and design method.

When the object oriented (OO-) modelling techniques came up in the 1980ies and 1990ies, the former E/R diagrams were enhanced to form so-called *class and object diagrams*. These encompass certain aspects of behaviour modelling which are expressed by operation definitions attached to the entities and their attributes. This development culminated in the Unified Modelling Language (cf. [22]) that offer class structure diagrams for conceptual modelling and other diagram techniques for modelling further aspects like dynamics, states and state transitions, co-operation of components etc.

Both approaches have in common the goal of representing the relevant "things" of an application area and their interrelationships. In principle, for defining an ontology the same concepts and languages may be used. In both fields we find a "syntactical" and a "semantical" region: The first encompasses the naming, classification and structural description of "things" (*entities*) and their associations (*relationships*), including their behaviour expressed by operations with their names, parameter and result types. Parts of the "semantics" are further conditions, constraints, connections and "meaning" – background information which often is not (conveniently) expressible by (foreground) diagrams or by formal language. A conceptual model is the more "semantic", the more it is able to cover such phenomena in a formalised way.

However, there are differences as well: First, we have to deal with different scopes. Traditional CM is normally used for one particular project (or at least for a given, known, limited family of projects), its concepts and definitions do normally not refer to areas beyond the project scope. Requirements on the commitment and stability of the model are limited to the project or project family scope, i.e. a conceptual model can be changed whenever the involved project members are informed and agree on it.

This is quite different for an ontology: The group of people (or even automata) who share its contents (in the sense of Gruber's Definition, cf. above) is much bigger and not limited to a specific project. It encompasses future projects and developments within the same domain including potential, possibly still unknown ontology use(r)s. The increased number of people from which an *ontological commitment* [9] is expected and the extension of such commitments to non-human devices are main reasons to demand for more precise, exchangeable and negotiable ontologies.

Spyns et al. contrast essential properties of data models and ontologies [21]:

- *Operational levels:* While data models contain domain rules expressed on a rather low, implementation-oriented level (e.g. concerning null values or primary keys) ontologies should be implementation-independent.
- *Expresssive power:* According to the different operational levels different sorts of languages and description tools are appropriate. SQL is a good language for data modelling but surely not for ontologies where the query aspect is less important than e.g. structural and taxonomical issues.

- *Users, purpose and goals:* Here the differences follow directly from the different scopes: Data models are defined for a specific application in a limited domain, while ontologies should span the borderline to adjacent domains and be usable for many applications (including unknown, future ones).
- *Extensibility:* With respect to this point the requirements on ontologies are much more challenging: When a data model is defined, likely extensions can be foreseen and taken into account. Other for an ontology: ideally it should anticipate non-foreseen uses and be open for any kind of extensions.

The authors highlight the apparent conflict between genericity and effectiveness: The more generic an ontology is conceived, the broader is its potential scope – but being less specific normally implies to abandon specific domain rules that are needed for effective interoperability (cf. [21], p.3).

4 Approaches to an ontology-based software process

If we want to characterise the specifics of *Ontology-based Software Engineering (OBSE)*, many questions come into play. First we try to list and answer a few principal ones of these and then discuss some alternatives for founding an OBSE process.

Q.: For which kind of projects could an ontology-based approach be adequate?

A.: Whenever a software development project is not just concerned with a specific application but is part of a landscape of projects located in the same domain, an ontology-based approach might be worth considering and even be advantageous to a traditional approach.

Q.: Which are the specific goals of OBSE (on contrast to traditional approaches)?

A.: One principal goal is to extend the idea of reuse from the implementation to the modelling level. I.e. instead of building systems from ready-made components which are "plugged together" like hardware modules, ontologies are *reusable model components* from which particular implementations can be derived for specific platforms, according to specific interfaces and constraints of application development projects.

Q.: Which concepts and paradigms of recent software technology do fit better or worse to an OBSE approach?

A.: Well-known recent paradigms are the *Model-Driven Development (MDD)* and the *Agile Software Development (ASD)* approach. MDD puts the model in the centre of the development process and thus encourages the reuse of models – as is required for OBSE. On the other hand, agile methods do not emphasise modelling activities or specifications but prefer quick implementation cycles and code refactoring. Therefore, in OBSE their place (and use) might be restricted to particular tasks like building prototype applications or implementations of ontologies.

Q.: Which languages and description tools are to be preferred for OBSE?

A.: This is still a very controversial question. Well-known ontology implementation languages like XML+RDF or DAML+OIL are on the wrong operational level – maybe good for automated agents but inappropriate for software engineers while

modelling. An ontology language in the SE context should primarily be made for humans - but hopefully be translatable for automata. Of course, UML is a serious candidate for an OBSE language. (cf. e.g. [1], [2]). However, it can be argued that it is too wide on the one hand side and too narrow on the other.

Too wide - since many of its constructs like sequence or deployment diagrams are out of scope or at least marginal from an ontological point of view. Too narrow - since the UML class structure is rather rigid and, for example, attributes, operations and associations are no first-class constructs, i.e. can only be used in connection with class definitions. This raises a further, more fundamental and well-known question concerning *granularity* in CM: Should one formally distinguish entities (classes/ objects) from features (attributes, operations) - as do the E/R- and UML approaches – or treat them all the same way, i.e. follow an ORM-like approach (cf. [11], [21])?

To answer these and further questions in detail would require much more space (and work) than can be spent for this paper. Instead, I shall concentrate on issues regarding the *OBSE process*. If we start from the traditional SE life cycle, which of its phases are mostly affected by the OBSE goals? Of course, all those that are particularly concerned with the application domain, i.e. the early phases of *system and requirements analysis* and of *domain modelling* and the late phases of *integration, deployment, use and revision* of the released software.

Following the now popular, UML-based process models, software development starts with analysing its requirements based on a use case structure *(use case analysis).* Use cases are mainly analysed and documented on a verbal, informal level using natural language (cf. [16]). In an ontology development project, the role of use cases might be played by *ontology application cases*, i.e. potential uses of the ontology by (present or future) application projects.

According to Jacobson, use cases can be exploited to find objects, list them in an *object list* and, based on that list, define the *analysis model*, i.e. a UML class structure diagram (or set of diagrams) complemented with the corresponding attribute and operation definitions. Such a procedure – which is already somehow weak for a "normal" development project – is not appropriate for OBSE. A promising approach to derive concepts from use case descriptions employs Formal Concept Analysis [G-W 98], [D-H 00]

The main purpose of an ontology is just to replace restricted, unstable object lists and analysis models by a more stable, well-founded and approved conceptual framework. Here we have to distinguish projects which imply developing a *new ontology* or at least enhancing it by new parts from those building *on an already existing* ontology. In the first case all relevant concepts, properties, relationships, states etc. of the concerned domain have to be defined in a common taxonomical framework. Normally, the scope and generality of the framework go beyond the immediate project requirements. In order to be general enough for further use, the ontology has to be negotiated not only with the immediate customer(s) but has to reflect previous work in the domain and to be confirmed by institutions responsible for that domain such as standardisation committees or taxonomists.

In the second case the project can (hopefully) profit from the OBSE approach. The existing ontology is used as basis for building the analysis model. Classes and objects of this model can be derived from and have to be anchored in the existing ontology. Relevant definitions of the ontology are specialised, adapted or extended according to the particular project requirements. New definitions have to be compatible with the ontology and be prepared for acceptance to it (provided they are general enough). This is the kind of *ontological commitment* to be expected from software projects in a domain covered by an ontology.

Which form is most appropriate for defining and communicating ontologies in software engineering projects? We have already briefly discussed the language level and benefits and shortcomings of a UML-like approach. The main arguments in favour of UML are its diffusion and diversity, opposed by the counter-arguments of wrong granularity, missing flexibility and logical rigidity (cf. above).

Another promising approach builds on glossaries: In the KCPM methodology, a *glossary* is employed as the central knowledge base for gathering, storing and communicating domain knowledge during the requirements capture and modelling phases of software projects [18]. The glossary is built up by two kinds of (table-like) type descriptions: *thing types* and *connection types*. In order to support the glossary building task, linguistic techniques such as natural language text analysis are employed and supported by corresponding tools [5]. Glossaries may be transformed into conceptual models or UML-like class structure diagrams according to a set of laws and transformation rules in a semi-automated way.

Spyns et al. follow a similar approach based on the ORM paradigma (*Objects with Roles-Model,* [11]). The authors distinguish a set of generally mandatory definitions called ontology base and packages of domain-specific rules called ontological commitments (cf. [21]). The *ontology base* consists of a collection of fact declarations, the so-called *lexons*, which basically express n-ary associations between terms (representing things or other associations). A *commitment* consists of a set of semi-formal rules – i.e. rules partly given in natural language, partly in a formal language. They mediate between the ontology base and its applications. For example, it is determined by such rules, which parts of the ontology are visible for the specific application and which are not or which additional conditions or constraints apply to them. In the course of application development projects, these rules can step-by-step be further formalised. Thus glossary-based modelling work is always performed in two steps:

(1) analysing and further refining the ontology base and

(2) adapting this base to the own project and to the specific application by ontological commitment, i.e. determining rules for their use, constraints or extensions.

To sum up, we can state: An ontology can well be used to facilitate software development processes in the long term horizon and serve for more homogeneous systems – in particular with respect to their interoperability and (partial) re-use. An ideal ontology should comprise domain knowledge as detailed and accurate as possible, but on the other hand it should be free from implementation-dependent details which might constrain the space of possible solutions in an undue way.

5 The ontology life cycle

Now we are ready to look for the ontology development process against the background of SE processes. First we consider an ontology as a special piece of "software" which has its own – mostly rather long-term life cycle. This life cycle is intertwined with the life cycles of software application development projects that are connected to the ontology by the same application domain. Both life cycles are maintained by different roles: the software engineer and the ontology engineer. Nevertheless Fernandez et al. have emphasised a certain analogy between the two processes and have investigated well-known SE life cycle models as potential paradigms for ontology development (cf. [4]). Among these are:

- Waterfall,
- incremental development,
- evolutionary development.

Obviously, waterfall-like models are inappropriate: In general, ontology development does not follow a sequential plan with consecutive phases, intermediate results, milestones etc – whereas particular subjects (*ontology components*) might be developed in such a way. More appropriate seems an incremental approach: starting with a kernel ontology, which is stepwise extended by "ontology increments" i.e. partial formalisations of particular subdomains. However, incremental development has to follow an overall plan as well, triggered by the "milestones" of released increments.

For ontology development, an evolutionary approach seems to be most appropriate: It might start with an "ontology prototype" comprising some basic definitions – maybe stated as an initial glossary. Then this is refined and enhanced step by step through a series of application development projects. This means: Each application project initiates a new ontology evolution cycle. For modelling evolution cycles, the EOS model (for *E*volutionary, *O*bject-oriented *S*oftware development, cf. [13], [15]) may be helpful. Among its key concepts are:

- *Component-based structure:* Complex systems are viewed as hierarchical compositions of parts called components and modules. Correspondingly, we decompose a (sufficiently complex) ontology into sub-ontologies which can be developed (relatively) independent from each other but have to be integrated and adapted to the sharing partners afterwards.

- *"Fractal" development cycles following the system architecture:* Unlike traditional life cycle models, EOS links development cycles with the system structure (and not vice versa): every component has its own cycle. Typically, these are performed concurrently and have to be synchronised – leading to a "fractal" process structure (cf. fig. 3, right part). Accordingly, ontology development can be viewed as a complex process structured in a "fractal" way, where single components may be worked out as parts of particular application development projects.

- *Homogeneous concurrent development cycles:* All development cycles (be it on the system, component or sub-component level) have the same structure consisting of the four main phases: *analysis*, *design*, *implementation* and *operational use*. In the context of ontology development, these phases can be characterised as follows:

Analysis: In case of building a new ontology, this first phase serves for starting the overall ontology development process. The universe of discourse, goals and purposes are determined, possible sources of knowledge are identified. Interfaces and references to other existing, relevant ontologies or components (prior, ancestor, descendent, neighbour, competing, conflicting ones ...) are investigated. The role of "use cases" is taken by existing or prospective applications, i.e. projects which (might) use the ontology presently or in the future. The level of formality, possible description languages, tools etc. are determined. A (provisional) glossary is built – e.g. following the KPCM or ORM approach (cf. above). An overall ontology development plan and a strategy for its further development and integration are defined. In case of re-working an existing ontology, analysis starts with the already existing definitions and builds on them.

Design: The structure and hierarchy, possible sub-ontologies, interfaces and rules for commitment are determined in this phase. The ontology domain is conceptualised, the glossary is (further) filled, enhanced, and extended, explanations and cross-references are given. Mappings and translations to "dialects" like E/R- or UML-diagrams or formal ontology languages are provided. Completeness and consistency of definitions are checked.

Implementation: The ontology is translated into a concrete ontology or programming language (e.g. DAML+OIL, OWL, XML+RDF, DL, Prolog, Java classes) according to the requirements. Results from sub-development cycles (regarding sub-ontologies) are integrated. The ontology definitions are published for (potential) users.

Operational use: The integration process is completed with ancestor and neighbour ontologies (cf. also [4], p. 2). Integration units are checked for overlapping parts, inconsistencies, incompleteness etc. Feedback from application projects, organisations, users, etc. is gathered and analysed. If justified by new requirements and the administrative prerequisites are given, a revision cycle may be started.

Now we are ready to compare the Software Engineering and Ontology Engineering life cycles. In the following table, some of their characteristics are contrasted:

	Software Engineering	**Ontology Engineering**
Duration	• determined, limited for one project	• undetermined, unlimited
Structure	• phases, grouped in iterations and activities	• evolution cycles, grouped in phases and activities
Sub-processes	• for components, increments or special tasks (e.g. QA)	• for developing or revising subdomains
Paradigms	• waterfall • incremental • component-based	• incremental • component-based • evolutionary

	• prototyping, spiral-like	
Activities and artefacts	• find and describe use cases • elicit requirements • build conceptual model / class structure • build system architecture, specify components and modules • code: use/generate concrete programming language • test and debug • integrate, test subsystems • deploy system • validate, adapt to environment • get feedback form users • start revision (if necessary)	• identify potential applications • analyse terminology • build taxonomy • build and fill glossary • define facts and rules • check with other glossaries, solve terminological conflicts • formalise: translate into formal ontology language • integrate: link and validate sub-ontologies, adapt to super-/ neighbour ontology • publish ontology • validate, receive feedback • elicit requirements for revision
Languages/ description tools	• use case diagrams • natural language • E/R diagrams, UML • pseudo code • programming language(s)	• natural language • E/R diagrams, UML • glossaries, tables • semantic networks • topic maps • logic languages • conceptual graphs • frames, DAML+OIL, OWL
Target groups	• (other) Developers • Users • Customers	• Domain experts • Developers • Agents of other systems
Results and products	• project-specific • short-term oriented • (mostly) not re-usable • isolated, restricted for particular application	• spanning many projects • long-term oriented • re-usable • "sharable" among many organisations and projects

Fig. 2: Software Engineering vs. Ontology engineering

6 Outlook: Perspectives of *Ontology-based Software Engineering*

In this last section, I want to resume my vision on *Ontology-based Software Engineering (OBSE)* in the form a few theses:

- Ontology development and software development have their own, concurrent, intertwined cycles which have something in common but also differ in their goals, responsibilities and time horizons. Ontology development is more long-term oriented and has a wider scope. Normally it is linked to many software projects and managed by project-spanning teams and organisations.
- If there exists already an ontology for the application domain of a software project, this is a candidate for an OBSE project. During the early phases it can "dock" at the ontology development and profit from earlier work done there. In the late phases, there is a backward knowledge transfer of the project results to the ontology – as far as they are relevant for it. The EOS model can be used as a basis for a common, co-evolutionary model (cf. fig. 3)

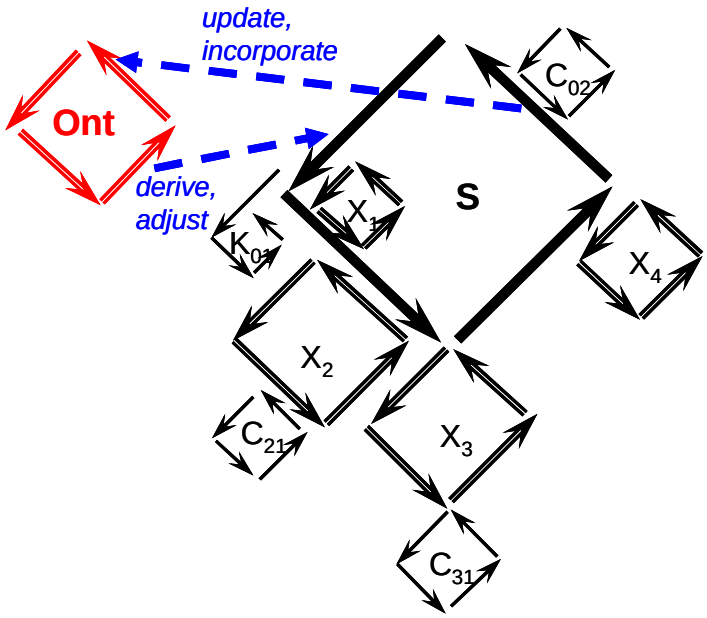

Fig. 3: Co-evolution of ontology and software development

- In a first attempt, the ontology might be viewed as a special (pre-fabricated) *component* in the software development process. However, if it comprises the whole application domain it is more likely to represent knowledge to be used in many components of the system to be developed. Thus the ontology is taken as a *"codified model"* and a starter version of the overall system analysis model. During the project, it is further developed and parts of it influence the development of particular components of the system.

- Ontology and software development are intertwined in manifold respects. Particularly important are the connections in the early and late phases:

  - *System analysis* implies the *investigation* of existing ontologies and the transfer of codified knowledge for the application domain being considered.

  - *System design* builds on ontological definitions and *commitments* for the current project.

  - *System implementation and operational use* imply *feedback* of the project *results* and *experiences* to the ontology developers/maintainers and have to be incorporated there in order to keep the ontology a living organism.

- Ontologies are a promising instrument for *knowledge transfer* from project to project in a certain application domain and from one development cycle of a project to the next. In the mid- and long-term future, OBSE might become an attractive software engineering paradigm which serves for closer co-operation, better compatible models, more re-usable components and less costs in the software development field.

References:

[1] K. Baclawski et al.: Extending the Unified Modeling Language for ontology development. Software and Systems Modeling - SoSym 1 (2), pp. 142-156 (2002)

[2] S. Cranefield: UML and the Semantic Web, Sem. Web Working Symposium, Stanford 2001 http://www.semanticweb.org/SWWS/program/full/paper1.pdf

[D-H 00] S. Düwel, W. Hesse: Bridging the gap between Use Case Analysis and Class Structure Design by Formal Concept Analysis. In: J. Ebert, U. Frank (Hrsg.): Modelle und Modellierungssprachen in Informatik und Wirtschaftsinformatik. Proc. "Modellierung 2000", pp. 27-40, Fölbach-Verlag, Koblenz 2000

[3] R. Dupuis, P. Bourque, A. Abran, J. W. Moore, and L. L. Tripp :The SWEBOK Project: Guide to the Software Engineering Body of Knowledge
Presented at ICSSEA '99 Paris, France December 8-10, 1999.

[4] M. Fernandez, A. Gomez-Perez, N. Juristo: Methontology: From ontological art towards ontological engineering, Symp. on Ontological Engineering of AAAI, Stanford Ca. (1997)

[5] G. Fliedl, Ch. Kop, H. C. Mayr, W. Mayerthaler, Ch. Winkler: Linguistically based requirements engineering - The NIBA project. In: Data & Knowledge Engineering, Vol. 35, 2000, pp. 111 - 120

[6] Ch. Floyd, H. Züllighoven, R. Budde, R. Keil-Slawik: Software Development and Reality Construction, Springer-Verlag 1992

[G-W 98] B. Ganter, R. Wille: Formal Concept Analysis, Mathematical Foundations, Springer 1998

[7] N. Guarino: Formal Ontology and Information Systems. In: Proc. FOIS '98, Trento (Italy) June 1998, Amsterdam IOS Press pp 3-15

[8] Gesellschaft für Informatik (GI)-Arbeitskreise "Terminologie der Softwaretechnik" und "Begriffe für Vorgehensmodelle": Informatik-Begriffsnetz. http://www.tfh-berlin.de/%7Egiak/

[9] T. Gruber: A translation approach to portable ontologies. Knowledge Acquisition, 5(2), pp. 199-220 (1993), also: *What is an Ontology?* http://www-ksl.stanford. edu/kst/what-is-an-ontology.html

[10] M. Gruninger, J. Lee: Ontology - Applications and Design. CACM 45.2, pp. 39-41 (Feb. 2002)

[11] T. Halpin: Information Modeling and Relational Databases: from conceptual analysis to logical design. Morgan-Kaufmann 2001

[12] D. Hensel: *Relating Ontology Languages and Web Standards*. In: J. Ebert, U. Frank (Hrsg.): *Modelle und Modellierungssprachen in Informatik und Wirtschaftsinformatik. Proc. "Modellierung 2000"*, Fölbach-Verlag, Koblenz 2000, pp. 111-128

[13] W. Hesse: Theory and practice of the software process - a field study and its implications for project management; in: C. Montangero (Ed.): Software Process Technology, 5th European Workshop, EWSPT 96, Springer LNCS 1149, pp. 241-256 (1996)

[14] W. Hesse: Das aktuelle Schlagwort: Ontologie(n). in: Informatik Spektrum, Band 25.6 (Dez. 2002)

[15] W. Hesse: Dinosaur Meets Archaeopteryx? or: Is there an Alternative for Rational's Unified Process? Software and Systems Modeling (SoSyM) Vol. 2. No. 4, pp. 240-247 (2003)

[16] I. Jacobson, G. Booch, J. Rumbaugh: The Unified Software Development Process. Addison-Wesley 1999

[17] P. Janich: Wozu Ontologie für Informatiker? Objektbezug durch Sprachkritik. In: K. Bauknecht et al. (eds.): Informatik 2001 - Tagungsband der GI/OCG-Jahrestagung, Bd. II, pp. 765-769. books_372ocg.at; Bd. 157, Österr. Computer-Gesellschaft 2001

[18] H.C. Mayr , Ch. Kop: A User Centered Approach to Requirements Modeling. In: M. Glinz, G. Müller-Luschnat (Hrsg.): Modellierung 2002 - Modellierung in der Praxis - Modellierung für die Praxis, pp. 75-86, Springer LNI P-12 (2003)

[19] J. Miller, J. Mukerji: MDA Guide. Version 1.1.1, Object Management Group 2003.

[20] A. Mädche, H.-P. Schnurr, S. Staab, R. Studer: Representation-Language-Neutral Modelling of Ontologies, in: J. Ebert, U. Frank (Hrsg.): *Modelle und Modellierungssprachen in Informatik und Wirtschaftsinformatik. Proc. "Modellierung 2000"*, pp. 143-150, Fölbach-Verlag, Koblenz 2000

[21] P. Spyns, R. Meersman, M. Jarrar: Data modelling versus Ontology engineering, SIGMOD Record 31 (4), Dec. 2002

[22] Unified Modeling Language (UML) 1.5 Documentation. OMG document ad/99/06-09. Rational Software Corp., Santa Clara, CA 2001. http://www.rational.com/uml/resources/documentation

Integrationsarchitektur der Credit Suisse (eingeladener Vortrag)

Claus Hagen

Credit Suisse, Dept. KXI, P.O.Box 600, CH-8070 Zürich
claus.hagen@credit-suisse.com

Abstract:

Finanzunternehmen sind traditionell sehr abhängig von ihrer IT-Applikationslandschaft. Während in anderen Branchen reale Güter produziert werden, existieren die Produkte der Banken und Versicherungen nur virtuell innerhalb der IT-Systeme. In der Finanzbranche setzt man schon seit den 70er-Jahren in grossem Stil Informationssysteme ein. Viele der Systeme aus den 70er und 80er-Jahren sind heute noch in Betrieb. Die typische Applikationslandschaft einer Grossbank stellt sich heute in der Regel als ein Konglomerat heterogener Applikationen aus verschiedensten Jahrzehnten dar, die mit unterschiedlichsten Technologien gebaut wurden. Diese Landschaften haben eine sehr hohe Komplexität, was ihre Grösse, ihre Heteroneität, und die gegenseitigen Abhängigkeiten zwischen ihren Komponenten angeht.

Gleichzeitig steigt die Frequenz, mit der neue Anforderungen an die Unternehmen gestellt werden. Die schnellebigen Finanzmärkte, der hohe Margendruck, und die steigenden Ansprüche der Kunden erfordern die möglichst rasche Einführung neuer Produkte, die flexible Anpassung an sich ändernde Märkte, und gleichzeitig die Reduktion der Produktionskosten. Da in der Finanzbranche jedes neue Produkt auch eine neue IT-Lösung bedeutet, gelten diese Anforderungen auch für die Informatik. Sie stellt einen wesentlichen Einflussfaktor auf die Flexibilität des Unternehmens und seine Kosten dar.

Hohe Flexibilität bei gleichzeitig niedrigen Kosten lässt sich auf der IT-Seite nur durch konsequentes Architekturmanagement erreichen. Wir verstehen darunter die Einführung und Weiterentwicklung unternehmensweiter Standards und Strukturregeln im IT-Bereich sowie deren Umsetzung.

Die Credit Suisse ist einer der grössten Finanzdienstleister der Schweiz und ist schon seit längerem mit den oben beschriebenen Herausforderungen konfrontiert. In der zweiten Hälfte der 90er-Jahre wurde bei der Credit Suisse ein durchgängiges Architekturmanagement eingeführt. Dieses umfasste von Beginn an eine wohldefinierte Integrationsarchitektur. Sie basierte zunächst auf einer reinen Service-Architektur, deren Etablierung die rasche Anbindung der alten, Mainframe-basierten Kernsysteme an neue Portale im Internet- und Intranet-Bereich ermöglichte. Heute sind ca. 800 Services in Produktion, die inzwischen ungefähr 30% des täglichen Transaktionsvolumens transportieren (bei steigender Tendenz).

Auf Basis der erfolgreichen Servicearchitektur wurde die Integrationsarchitektur der Credit Suisse kontinuierlich weiterentwickelt. Heute existiert eine umfangreiche Palette von Integrationsmechanismen, die die flexible Kombination der

unterschiedlichen Applikationssysteme ermöglicht und so die rasche Reaktion auf neue Anforderungen ermöglicht. Alle Mechanismen basieren jedoch einheitlich auf den Prinzipien der Kapselung und der Kommunikation „by Contract“.

Im Vortrag stellen wir die Grundelemente und –Prinzipien der Integrationsarchitektur der Credit Suisse vor. Wir berichten über die gemachten Erfahrungen und die „Lessons Learned“. Schliesslich geben wir einen Ausblick auf derzeit laufende und geplante Arbeiten.

Wissenschaftliches Programm

Anwendungsintegration im Krankenhaus am Beispiel eines interdisziplinären Tumorkolloquiums

Dirk Schröder [a], Martin Middeke [b], Klaus Kuhn [c], Richard Lenz [c]

Klinikum der Philipps-Universität Marburg
[a] Zentrale Informationsverarbeitung
[b] Marburger interdisziplinäres Tumorzentrum[1]
[c] Institut für Medizinische Informatik

Der Behandlungsprozess eines Patienten im Krankenhaus ist sehr stark durch interdisziplinäre arbeitsteilige Abläufe geprägt. Ein reibungsloser Informationsfluss zwischen den beteiligten Personengruppen ist von entscheidender Bedeutung für die Gewährleistung einer hohen Behandlungsqualität. Der interdisziplinären Tumorbehandlung kommt hier ein erheblicher Stellenwert zu. Im vorliegenden Artikel wird am Beispiel des interdisziplinären Tumorkolloquiums am Universitätsklinikum in Marburg aufgezeigt, wie derartige Abläufe mit Hilfe dokumentenbasierter IT-Anwendungen unterstützt werden können.

Hintergrund und Motivation

Im Rahmen der Patientenbehandlung im Krankenhaus wird an vielen Orten von zahlreichen Beteiligten patientenbezogene Information erfasst, die in den korrekten Kontext zu bringen und in geeigneter Form am „point of care“ bereitzustellen ist. Eine geeignete IT-Unterstützung zur Optimierung des Informationsflusses in derart interdisziplinären Abläufen erfordert integrierte, aber gleichwohl erweiterbare Systeme, die an ständig neue Anforderungen flexibel angepasst werden müssen [1]. Ziel der Integration ist der nahtlose Informationsfluss über Abteilungsgrenzen, idealerweise auch Institutionsgrenzen hinweg, die Vermeidung von Mehrfacheingaben, die Minimierung von Inkonsistenzen durch Reduzierung unkontrollierter Redundanz, die möglichst einfache Wiederverwendbarkeit von Informationen im Rahmen vielfältiger Anwendungen sowie die Herstellung einheitlicher Konzepte zur Benutzerinteraktion (Single System Image). Am Universitätsklinikum in Marburg (UKMR) wurde mit am Markt verfügbaren IS - Komponenten eine responsive IT-Infrastruktur aufgebaut, auf deren Basis nun mit Hilfe eines Generatorwerkzeugs („Composer“) für formularbasierte klinische Anwendungen ein kontinuierlicher Ausbau der klinischen Funktionalität stattfindet [2;3]. Im vorliegenden Papier wird die Nutzung dieser Infrastruktur für abteilungsübergreifende Anwendungen diskutiert (vgl. auch [4]). Dazu wird exemplarisch die Implementierung einer IT-Unterstützung für das Marburger Tumorkolloquium vorgestellt.

[1] gefördert durch die Deutsche Krebshilfe

Die moderne Behandlung von bösartigen Tumoren erfordert eine enge interdisziplinäre Zusammenarbeit zwischen allen onkologischen Fachrichtungen. Hierzu zählen nicht nur die Fachdisziplinen Hämatologie/Onkologie, Strahlentherapie und Chirurgie mit ihren Spezialgebieten in der Gynäkologie, Dermatologie, Urologie etc., sondern auch in nicht geringerem Maße die allgemeineren Disziplinen wie Pathologie und Radiologie. Diese große Anzahl an Fachdisziplinen in der interdisziplinären Zusammenarbeit zu organisieren, gelingt am besten in regelmäßigen Tumorkolloquien, an denen Vertreter der einzelnen Disziplinen persönlich teilnehmen [5;6]. Für die sachgerechte, patientenbezogene Empfehlung aus der jeweiligen Fachrichtung ist eine vorherige fundierte Information über den Erkrankungsverlauf des individuellen Patienten zwingend erforderlich. Darüber hinaus ist die patientenbezogene Dokumentation der Empfehlung in der elektronischen Patientenakte für die abteilungsübergreifende Behandlung dringend erwünscht.

Zielsetzung

Die Zielsetzung dieses Artikels besteht zunächst darin, die Praktikabilität der in Marburg gewählten IT-Infrastruktur im Hinblick auf die flexible Erweiterbarkeit aufzuzeigen bzw. auf der Basis der Marburger Projekterfahrungen kritisch zu hinterfragen. Der hier vorgestellte Prozess des Marburger Tumorkolloquiums zeigt einen abteilungsübergreifenden Anwendungsfall auf, an dem die Vorteile der Systemintegration besonders deutlich werden.

Methoden

Die Systemarchitektur des Krankenhausinformationssystems (KIS) am UKMR wird in Abb. 1 übersichtsartig dargestellt. (vgl. auch [7]). Das Kernsystem basiert auf dem Krankenhausinformationssystem ORBIS der Firma GWI. Es zeichnet sich durch eine zentrale Datenbank aus, die im Sinne des typischen ERP-Ansatzes als Integrationsplattform für unterschiedliche Anwendungen genutzt wird. Über parametrierbare Standardmodule hinaus ermöglicht das Programmierframework OpenMed dem Anwendungsentwickler auf einfache Art und Weise, das Kernsystem durch eigene formularbasierte Anwendungen zu erweitern. Die Applikationslogik und die zugehörigen Datenstrukturen werden dabei mit Hilfe eines graphischen Entwicklungswerkzeugs entworfen. Eine Erweiterung des Datenbank-Schemas ist dazu nicht erforderlich, da die Formularinhalte auf generische Datenbanktabellen abgebildet werden (vgl. [7]). Die werkzeugbasierte Anwendungsentwicklung beruht im Kern auf einem dokumentenorientierten Interaktionsparadigma, wobei Workflows durch die zustandsabhängige, dynamische Bereitstellung von elektronischen Dokumenten in verschiedenen Arbeitslisten (oder allgemein: „Referenzlisten“) unterstützt werden. Ein einheitlicher Zugang zum System wird durch benutzer-, rollen-, abteilungs- oder patientenspezifische Referenzlisten erreicht. Der Zugang zu Daten, die in autonomen Subsystemen erfasst werden (z.B. Labordaten), wird durch den Import über Standardschnittstellen (HL7) erreicht. Alle in der zentralen

Datenbank verfügbaren oder referenzierbaren Daten, sind im Rahmen elektronischer Formulare wiederverwendbar.
Aus Datenschutzgründen darf der Zugriff auf patientenbezogene Daten nur bei Vorliegen eines konkreten Behandlungszusammenhangs erfolgen. Eine entsprechende Einschränkung der Zugriffsrechte wird über ein rollenbasiertes, hierarchisches Rechtemodell sichergestellt, das ins Kernsystem integriert ist. Neuentwickelte formularbasierte Anwendungen nutzen ohne zusätzliche Aufwände das vorhanden Rechtemodell und sind somit datenschutzkonform integriert.

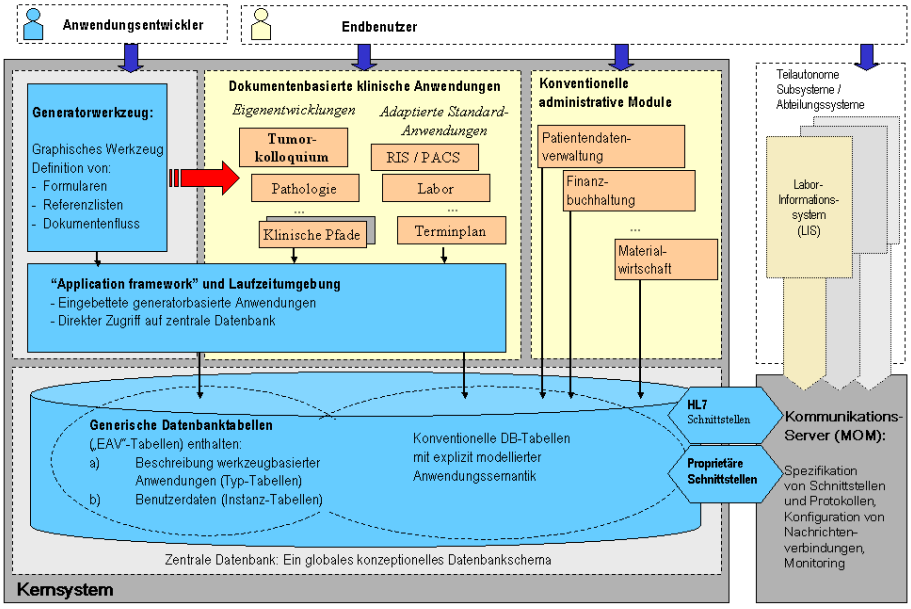

Abb. 1: Systemarchitektur des KIS am UKMR

Die werkzeuggestützte Anwendungsentwicklung erfordert ein entsprechend angepasstes Vorgehensmodell für den Softwareentwurf. Das traditionelle Wasserfallmodell ist ungeeignet, da es die im klinischen Umfeld typischerweise rasch wechselnden Anforderungen nicht berücksichtigt. Ein flexibles Vorgehensmodell mit kurzen Iterationszyklen wurde in [8] vorgestellt. Kern dieses Modells ist die Einbettung der Software-Entwicklung in einen kontinuierlichen Prozess der Geschäftsprozessoptimierung, sowie eine partizipative Vorgehensweise mit sehr kurzen Iterationszyklen. Dabei werden vor allem Aspekte des Change Managements in den Prozess integriert, um das Risiko für Projektfehlschläge zu minimieren. Als wesentliche Aspekte, die auch im konkreten Projekt *Tumorkolloquium* berücksichtigt wurden, sind herauszustellen: Die Sicherstellung des tatsächlichen Bedarfs, die Bestimmung eines Key Users als zentralem Ansprechpartner und die Gewährleistung der Unterstützung des Managements (Entscheidungs- und Handlungskompetenz).

Ergebnis

Ausgehend von der ausführlichen Spezifikation durch den Key User wurde ein erstes Konzept zur Realisierung entwickelt. Durch Rapid Prototyping konnten früh erste Entwürfe vorgestellt werden, die dann in enger Zusammenarbeit in kleinen Iterationsschritten verfeinert wurden. Für den Testbetrieb, der im Juli 2004 begann, wurden bewusst geringe, aber technisch beherrschbare, Einschränkungen akzeptiert. Der Key User war in der anschließenden Testphase für die Endanwender der Ansprechpartner und organisierte die notwendigen Schulungen. Alle Fehlermeldungen und Änderungswünsche wurden über den Key User zentral gesammelt und bewertet.
Da neben dem interdisziplinären Tumorkolloquium drei weitere Tumorkolloquien am UKMR existieren, die auch elektronisch unterstützt werden sollen, wurde aus Gründen der Effizienz, der Einheitlichkeit und des Wartungsaufwands ein wiederverwendbarer Ansatz gewählt. Dabei wird primär eine generische Anwendung geschaffen, welche zentrale Funktionen vorhält, die in allen Tumorkolloquien genutzt werden. Die konkreten tumorkolloquiumspezifischen Formulare können dann einfach in Subformularparametrierung erstellt und eingebunden werden.
Zentrale Funktionen wie der Zugriff auf Patientendaten, Diagnosen, Befunde, Labordaten, Termine und das RIS (Radiologisches Informationssystem), einschließlich PACS (Picture Archiving and Communication System), wurden über Standardfunktionen der Entwicklungsumgebung als Bestandteil der generischen Anwendung eingebunden. Tumorkolloquiumspezifische Formulare bilden die semantische Spezialisierung der einzelnen Tumorkolloquien ab.

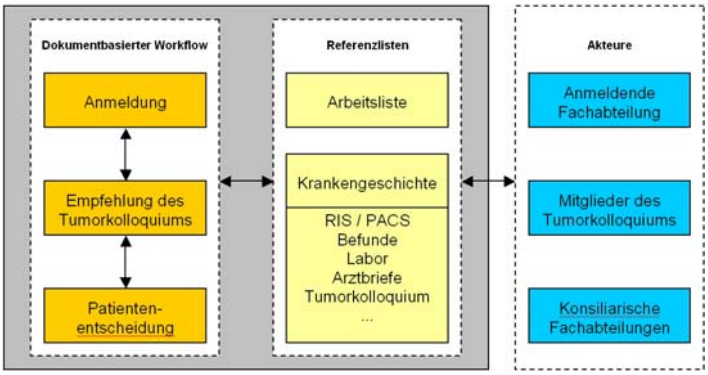

Abb. 2.: Formulare, Referenzlisten und Akteure im Workflow Tumordokumentation

Der Workflow wird durch die Anmeldung zum gewünschten Tumorkolloquium initiiert (Abb. 2). Der anfordernde Arzt wählt über das Standardmodul Terminbuch bereits bei der Anmeldung das Vorstellungsdatum des Patienten verbindlich aus. Datenschutzkonform können die das Kolloquium ausrichtenden Fachabteilungen jetzt auf die relevanten Patientendaten zugreifen. Die ggf. involvierten konsiliarischen Fachabteilungen können dagegen nur auf die Patienten zugreifen, zu denen ein entsprechender Behandlungszusammenhang definiert ist. Der Zugriff auf die Patienten erfolgt zeitlich befristet über eine gefilterte Arbeitsliste (Abb. 3). Nach der

Vorstellung des Patienten und der Beschlussfassung durch das Kolloquium wird die Empfehlung des Tumorkolloquiums im KIS bereitgestellt. Da die Empfehlungen wie alle anderen wichtigen Befunde in der zentrale Krankengeschichte des Patienten abgelegt werden, können jetzt alle Abteilungen mit Zugriff auf diesen Patienten diese Information elektronisch abrufen. Optional kann z.B. bei alternativen Behandlungsmethoden bzw. bei Ablehnung der Behandlung dieser Sachverhalt im optionalen Formular Patientenentscheidung dokumentiert werden.
Da die beschriebene Anwendung über das Laufzeitsystem des Kernsystems in das KIS integriert ist, können im Rahmen der Anmeldung zum Tumorkolloquium problemlos im System dokumentierte Daten wie z.B. Vorbefunde, Laborwerte oder Diagnosen übernommen werden. Seit der Teststellung eines PACS ist es darüber hinaus möglich, direkt aus dem Anforderungsformular über die dort zur Demonstration markierten radiologischen Aufnahmen in das PACS zu verzweigen, um diese Bilder (Röntgen, CT, MRT) während des Kolloquiums online zu visualisieren. Auf diese Weise entfällt die zeitaufwändige analoge Bereitstellung der radiologischen Aufnahmen.

Abb. 3: Arbeitsliste, Anmeldung zum Tumorkolloquium und Aufruf des PACS

Ein weiterer Vorteil der Integration in das Kernsystem besteht darin, dass nicht nur im Rahmen des Tumorkolloquiums auf zentral verspeicherte Daten zugegriffen werden kann, sondern dass auch die Ergebnisse des Tumorkolloquiums in anderen Dokumenten wiederverwendet werden können. Zeitsparend und fehlervermeidend kann so z.B. die Empfehlung des Tumorkolloquiums zur weiteren Behandlung des Patienten mühelos in den Entlassarztbrief übernommen werden.
Der Arbeitsaufwand für die Erstellung des Frameworks einschließlich der Subformulare für das interdisziplinäre Tumorkolloquium vom Vorliegen der Spezifikation bis zum Beginn des Testbetriebs lag bei ca. 25 Manntagen. Ein Teil dieser Aufwände kann als Einarbeitungszeit in spezielle notwendige Techniken angerechnet werden. Die zusätzlich notwendigen Arbeiten zur Erweiterung um ein weiteres Kolloquium beschränken sich lediglich auf die Parametrierung der spezifischen Subformulare mit ihren Katalogen, die Erweiterung des zentralen Katalogs, Anlage eines Terminkalenders, sowie die Administration des entsprechenden Systemverwaltungsformulars. Am 10.12.2004 wurde das Senologische

Tumorkolloquium des Marburger Brustzentrums Regio integriert. Die Aufwände für die reine Implementierung der Funktionalität lag hier bei ca. 5 Manntagen.
Seit Einführung der Organisation der interdisziplinären Tumorkolloquien ins KIS am 1.7.2004 sind (Stand 3.3.2005) 74 Kolloquien mit insgesamt 404 Patienten durchgeführt und dokumentiert worden. Die Zahl der im Tumorkolloquium vorgestellten Patienten hat sich im Vergleich zu einem Zeitpunkt vor der Einführung der elektronischen Anmeldung nicht verändert, sie liegt bei durchschnittlich 5-9 Patienten pro Kolloquium.
Durch die ins KIS integrierte Anwendung konnte die Anmeldung zum Tumorkolloquium für die beteiligten Ärzte erheblich vereinfacht werden: Zuvor wurden die notwendigen Befunde manuell zusammengestellt und zur Demonstration auf Folien kopiert. Eine individuelle Vorbereitung der Kolloquiumsteilnehmer war daher nicht möglich. In der IT-unterstützten Lösung können die Unterlagen einfach und flexibel zusammengestellt werden und sind für die Kolloquiumsteilnehmer mit dem Zeitpunkt der Anmeldung abteilungsübergreifend verfügbar.
Die Zahl der am Tumorkolloquium teilnehmenden Ärzte ist gegenüber der Zeit vor Einführung der IT-unterstützten Lösung deutlich gestiegen. Dieser Umstand lässt sich mit dem erhöhten Lerneffekt durch die Visualisierung der PACS-Bilder und die Möglichkeit zur Vorbereitung auf die speziellen Themen des kommenden Tumorkolloquiums erklären.

Diskussion

Die vorgestellte responsive IT-Infrastruktur des UKMR basiert im Wesentlichen auf einem holistischen Ansatz mit zentraler Datenbank. Der Einsatz eines integrierten Generatorwerkzeugs für formularbasierte Anwendungen unterstützt einen partizipativen Software-Entwicklungsprozess mit kurzen Iterationszyklen und damit eine anforderungsorientierte Systemevolution. Die Praktikabilität dieser Evolutionsstrategie konnte am Beispiel des interdisziplinären Tumorkolloquiums gezeigt werden. Als wesentlicher Erfolgsfaktor ist dabei vor allem die intensive Zusammenarbeit mit dem verantwortlichen Key User hervorzuheben, da er in das Projekt sowohl fachliche Kompetenz und Entscheidungsbefugnis, als auch zeitliche Freiräume zur aktiven Umsetzung einbrachte. Im klinischen Umfeld sind diese Voraussetzungen aufgrund der dauerhaft angespannten Personalsituation und der ausgeprägten Hierarchie leider nicht immer gegeben.
Die Integration in das Kernsystem wird sowohl bei den Entwicklern als auch bei den Anwendern als wesentlicher Vorteil der holistischen Evolutionsstrategie wahrgenommen. Weder für die Übernahme von Vorbefunden und patientenbezogenen Dokumentationen aus unterschiedlichsten Abteilungen, noch für die Bereitstellung der Empfehlungen des Tumorkolloquiums zur automatischen Übernahme in nachfolgende elektronische Dokumente war zusätzlicher Integrationsaufwand erforderlich. Bei einer alternativen Strategie mit autonomen Anwendungskomponenten hätte diese Aufgabe den Kern des Implementierungsaufwands ausgemacht. Der beschriebene Ansatz hat es ermöglicht, in einem überschaubaren Zeitraum mit begrenzten personellen Ressourcen eine zentrale Anwendung zu realisieren.

Die graphischen Möglichkeiten zur Modellierung und Dokumentation von Workflows sind in der vorliegenden Version des Generatorwerkzeug noch eingeschränkt, so dass beim Übergang von der Spezifikation auf die Implementierung ein Medienbruch im Softwareentwicklungsprozess entsteht. Der Hersteller ist sich dieses Problems bewusst und arbeitet an einer in das Generatorwerkzeug integrierten Engine zur Modellierung, Implementierung und Ausführung von Workflows.
Die Integration der ausstehenden zwei interdisziplinären Tumorkolloquien wird von den beteiligten Abteilungen gewünscht. Um die Behandlung von krebskranken Patienten durch IT-Unterstützung weiter zu optimieren, wird derzeit intensiv an einer ins KIS integrierten dokumentenbasierten Tumordokumentation gearbeitet.

Literaturverzeichnis

[1] Lenz R, Kuhn KA. Aspekte einer prozessorientierten Systemarchitektur für Informationssysteme im Gesundheitswesen. In: Dadam P, Reichert M (Hrsg.): Informatik 2004 - Informatik verbindet, Band 2, Beiträge der 34.Jahrestagung der Gesellschaft für Informatik e.V.(GI); 2004: 530-536.

[2] Kuhn KA, Lenz R, Elstner T, Siegele H, Moll R. Experiences with a generator tool for building clinical application modules. Methods Inf Med 2003; 42(1): 37-44.

[3] Beyer M, Kuhn KA, Meiler C, Jablonski S, Lenz R. Towards a flexible, process-oriented IT architecture for an integrated healthcare network. In: Proceedings of the 2004 ACM Symposium on Applied Computing (SAC); ACM, 2004: 264-271.

[4] Lenz R, Büssecker F, Herlofsen H, Hinrichs F, Zeiler T, Kuhn KA. Demand-driven evolution of IT-systems in healthcare - a case study for improving interdisciplinary processes. Methods Inf Med.
Im Druck.

[5] Neubauer A, Wolf M, Engenhart-Cabillic R, Rothmund M. Funktion und Aufgabe eines interdisziplinären Tumorzentrums . Dtsch Med Wochenschr 2002;(127): 901-906.

[6] Santoso JT, Schwertner B, Coleman RL, Hannigan EV. Tumor board in gynecologic oncology. Int J Gynecol Cancer 2004; 14(2): 206-209.

[7] Lenz R, Elstner T, Siegele H, Kuhn KA. A practical approach to process support in health information systems. J Am Med Inform Assoc 2002; 9(6): 571-585.

[8] Lenz R, Kuhn KA. Towards a continuous evolution and adaptation of information systems in healthcare. Int J Med Inf 2004; 73(1): 75-89.

EAI im Krankenhaus – Ein Erfahrungsbericht zur Koppelung von SAP IS-H mit dem Klinischen Arbeitsplatzsystem ORBIS

S. Langenberg[1], M. Uerlich[2], M. Neugebauer[1] und C. Schneider[1]

[1] Zentralbereich für Information und Steuerung,
[2] Zentralbereich für klinisches Prozessmanagement,
Universitätsklinikum Bonn, Sigmund-Freud-Str. 25, 53105 Bonn

Zusammenfassung. Am Universitätsklinikum Bonn wurde eine Koppelung des SAP Systems mit dem Klinischen Arbeitsplatzsystem ORBIS über eine asynchrone Schnittstelle realisiert. Wichtige Erfolgsfaktoren für ein solches Projekt sind neben der Softwareentwicklung die Abstimmung der Basisdaten und der Berechtigungen. Der Betreuungsaufwand einer solchen Schnittstelle ist allerdings erheblich.

1 Einleitung

Während in der Vergangenheit der IT-Einsatz im kaufmännischen Bereich im Vordergrund stand, wird durch die verstärkten Bemühungen zur Kostensenkung, Effizienz- und Qualitätssteigerung im Gesundheitswesen auch der medizinische Bereich im Krankenhaus immer stärker von IT-Systemen durchdrungen [1].

Angesichts der Einführung eines pauschalierten Entgeltsystems (DRGs) zum 01.01.2004 für die Abrechnung im stationären Krankenhausbereich entschied sich der Vorstand des Universitätsklinikums Bonn (UKB), das administrative System SAP R/3 durch ein Klinisches Arbeitsplatzsystem (KAS) für den medizinischen Bereich mit folgender Zielsetzung zu ergänzen:

- Sicherstellung der vollständigen Dokumentation von Diagnosen und Prozeduren, die Grundlage der Abrechnung nach DRGs sind, insbesondere auch der fachabteilungsübergreifenden fallbezogenen Sicht auf die medizinische Dokumentation.
- Verbesserung der Koordination medizinischer Abläufe, wie z.B. der OP-Planung und Verringerung der Durchlaufzeit der Patienten.
- Erhebung von Leistungsdaten für eine zukünftige Kostenträgerrechnung und die Steuerung medizinischer Prozesse.

Es ergab sich nun die Problematik, das bereits seit 1999 eingeführte System SAP R/3, speziell mit den Modulen IS-H (Patientenmanagement und Abrechnung), FI (Finanzbuchhaltung), CO (Kostenrechnung) und MM (Materialwirtschaft), mit dem KAS über entsprechende Schnittstellen zu verknüpfen. Je nach Verlagerung des Patientenmanagements auf das KAS oder auf SAP IS-H sind

zahlreiche Varianten denkbar. Unter Beachtung der Zielsetzung, dass jede Berufsgruppe möglichst nur mit einem System arbeiten sollte, kristallisierten sich dabei zwei Schnittstellenvarianten heraus:

1. Ablösung des Moduls IS-H und Übernahme dieser Funktion durch das KAS. Die Funktion des patientenführenden Systems geht hierbei von SAP auf das KAS über. Bei dieser Konstellation wird eine Schnittstelle zum Modul FI von SAP benötigt, um die Rechnungen verbuchen zu können.
2. Beibehaltung des Moduls IS-H, hierbei müssen zwei patientenführende Systeme über eine komplexe Schnittstelle miteinander gekoppelt werden. Über diese Schnittstelle müssen Patientenstamm- und Bewegungsdaten bidirektional, medizinische Daten unidirektional vom KAS zu SAP übermittelt werden.

Von Seiten der DFG-Gutachter im Rahmen des HBFG-Verfahrens wurde ein eindeutiges Votum zugunsten der 1. Variante ausgesprochen, da die Komplexität dieser Schnittstelle deutlich geringer ist. Zudem ist die Kommunikation hier zeitunkritisch. Dennoch hat sich das UKB zur Realisierung der zweiten Schnittstellenvariante entschlossen, wobei folgende Gründe ausschlaggebend waren:

- Zum Zeitpunkt des ursprünglich geplanten Produktivstarts des KAS am 08.12.2003 konnte keiner der KAS-Anbieter, die im Rahmen der europaweiten Ausschreibung in die engere Wahl gekommen waren, ein Patientenmanagement- und Abrechnungssystem anbieten, das den gleichen Funktionsumfang wie das bereits für die UKB spezifische Abrechnung ambulanter und stationärer Leistungen eingeführte Modul IS-H geboten hätte.
- Die zweite Schnittstellenvariante bietet grundsätzlich die Option, zu einem späteren Zeitpunkt auf die erste Variante umzusteigen.
- Die Funktionsfähigkeit der administrativen Systeme, insbesondere die Abrechenbarkeit von Leistungen, musste sichergestellt werden, u.a. unter Berücksichtigung der zeitlichen Abwicklung des HBFG-Verfahrens, der EU-weiten Ausschreibung und der Vertragsgestaltung.

Die europaweite Ausschreibung zum Jahreswechsel 2002/2003 konnte die Firma GWI mit dem Produkt ORBIS/OpenMed, die zuvor schon am Universitätsklinikum Köln den Zuschlag für die Realisierung eines KAS erhalten hatte, für sich entscheiden. In Köln entschied man sich ebenfalls für die Schnittstellenvariante 2 unter Beibehaltung des Moduls IS-H [2]. Allerdings wurde dort das Modul IS-H bereits zuvor auch im medizinischen Bereich flächendeckend für das Patientenmanagement eingesetzt, so dass hier SAP das primär patientenführende System darstellt.

Das KAS-Projekt startete nach Abschluss des Vertrages mit der Firma GWI am UKB am 01.05.2003. Die Firma GWI übernahm für die Realisierung der Schnittstelle die Generalunternehmerschaft. Dem UKB kam dabei die Rolle des Pilotkunden zu. Der Produktivstart von ORBIS und der Schnittstelle war zunächst für den 08.12.2003 geplant, musste aber mehrfach wegen erheblicher Schwierigkeiten bei der Entwicklung der Schnittstelle verschoben werden. Er

erfolgte dann erst zum 05.07.2004, wobei noch nicht alle Schnittstellenkomponenten voll funktionsfähig waren. Die Realisierung einiger Komponenten ist noch in der Entwicklung.

2 Systemzuständigkeiten für Geschäftsprozesse

Geschäftsprozess	**SAP**	**ORBIS**
Patientenbewegungen		
Stationäre Aufnahme (Ist)	H	N
Stationäre Aufnahme (Plan)	N	H
Kurz-/ Notaufnahme	N	H
Ambulante Aufnahme	H	N
Verlegung, Entlassung	N	H
§301 Daten		
Diagnosen, Prozeduren	–	H
EDIFACT §301 Übermittlung	H	–
Datenpflege und Korrektur		
Personenstammdaten	H	N
Falldaten	H	–
Fallartwechsel ambulant → stationär	N	H
Fallartwechsel stationär → ambulant	H	–
Fallstatuswechsel Plan → Ist	H	N
Fallstorno	H	–
Storno Besuche	H	N
Personen zusammenführen	H	–
Stammdaten und Kataloge		
Organisationseinheiten	H	–
Zimmer und Betten	–	H
Kostenstellen	H	–
Externe Ärzte, Kostenträger und Krankenkassen	H	–

Tabelle 1. Verteilung der **H**aupt- und **N**ebenverantwortlichkeiten für die Geschäftsprozesse auf die Systeme SAP und ORBIS.

Die Verteilung der Geschäftsprozesse zwischen den Systemen ORBIS und SAP erfolgte unter der Prämisse, dass die Verwaltungsmitarbeiter nach wie vor ausschließlich mit SAP, das medizinische Personal ausschließlich mit ORBIS arbeiten sollte. Da Funktionen des Patientenmanagements von beiden Berufsgruppen genutzt werden, müssen diese Funktionen in beiden Systemen zur Verfügung stehen, siehe Tab. 1.

Die Anbindung des Moduls CO von SAP ist derzeit noch nicht realisiert, später sollen aus ORBIS direkt ohne Umweg über IS-H Leistungsdaten an SAP CO kommuniziert werden. Die Pflege von Stammdatenkataloge wie externen Ärzten, Krankenkassen und Krankenhäusern soll ausschließlich in IS-H erfolgen.

3 SAP IS-H – ORBIS Schnittstelle

3.1 Realisierungsvoraussetzung

Die allgemein notwendigen Voraussetzungen zur bidirektionalen schnittstellentechnischen Koppelung zweier Anwendungssysteme sind:

- Die Datenbankstrukturen beider Systeme sind in beide Richtungen aufeinander abbildbar. Um dies zu gewährleisten, musste das Datenmodell von ORBIS erweitert werden.
- Eine Transaktion, die in dem Quellsystem erlaubt ist, muss auch in dem Zielsystem durchführbar sein. Dies ist u.a. durch entsprechende Berechtigungsvergabe in beiden Systemen, sowie Anpassungen von Datenbankconstraints in ORBIS sicherzustellen.
- Werteausprägungen von Stammdatenkatalogen (Organisationseinheiten, Namenszusätze, Familienstände) müssen sich jeweils eindeutig von dem Quellsystem auf das Zielsystem abbilden lassen. Pflichtfelder müssen in beiden Systemen gleich sein. Dies ist durch einen sorgfältigen Basisdatenabgleich einerseits und durch Erweiterungen von ORBIS andererseits sicherzustellen.

3.2 Technische Realisierung

Die bidirektionale Koppelung von SAP mit ORBIS wird über eine asynchrone Kommunikation realisiert. Prinzip: *Send and forget*. Dies bedeutet, dass wenn eine Nachricht vom Zielsystem nicht verarbeitet werden kann, eine Dateninkonsistenz zwischen dem Quell- und Zielsystem vorliegt.

Auf eine synchrone Koppelung, die diese Konsistenzprobleme vermieden hätte, wurde verzichtet, da einerseits der Entwicklungsaufwand deutlich höher ist, andererseits eine zu starke Abhängigkeit der Verfügbarkeit von ORBIS von der Verfügbarkeit von SAP bestanden hätte. Daher müsste im Falle der Nichtverfügbarkeit von SAP, von synchroner Kommunikation auf asynchrone Kommunikation umgeschaltet werden können, wie dies beispielsweise in einem ähnlichen Szenario am Universitätsklinikum Leipzig bereits realisiert wurde [3].

Zur Kommunikation von SAP zu Fremdsystemen stellt SAP für die ADT-Kommunikation (Patientenstamm- und Bewegungssätze) das Hospital Communication Modul (HCM), ein proprietäres, ereignisorientiertes Nachrichtenformat, das Bestandteil des Moduls IS-H von SAP ist, zur Verfügung [4]. Das HCM-Format musste allerdings um kundenspezifische Nachrichten und Segmente erweitert werden (Neugeborenendaten, Fall-Fall Zuordnung, Patientenabwesenheit, Notaufnahmeart). Nach Ausführung einer Transaktion mit Änderung von patientenmanagementrelevanten Daten werden HCM-Ereignisse erzeugt. Durch den HCM-Nachrichtenaufbereiter, der in periodischen Intervallen von 6 min von SAP aufgerufen wird, werden die gesamten Ereignisse abgearbeitet und die zu den Ereignissen gehörenden HCM-Nachrichten erstellt. Die von SAP erzeugte HCM-Datei wird mit dem Kommunikationsserver e*Gate in ein ORBIS-spezifisches HL7-Format [5] verwandelt, das von dem Schnittstellenmodul JAIF von ORBIS weiterverarbeitet wird.

Die wichtigste Technik zur Kommunikation von Fremdsystemen zu SAP sind die sog. BAPIs. Sie bilden eine standardisierte objektorientierte Programmierschnittstelle zu den Business-Objekten des SAP-Systems [6] und kommen auch bei der Anbindung von ORBIS zur Anwendung. BAPIs sind als Methoden der SAP-Business-Objekttypen bzw. SAP-Interfacetypen definiert und werden als Funktionsbausteine implementiert. Da die Standard-BAPIs nicht ausreichend sind, wurden von GWI zusätzliche Funktionsbausteine (Anlage eines Versicherungsverhältnisses, Fallstatuswechsel, Neugeborenes etc.) implementiert.

In ORBIS wurde ein zu SAP IS-H vergleichbarer Eventmechanismus implementiert, diese Events werden vom Schnittstellenmodul JAIF in Nachrichten umgewandelt. Bei der ADT-Rückkommunikation wurden zunächst die BAPI-Aufrufe aus dem Kommunikationsserver heraus ausgeführt, dieser verarbeitet entsprechende HL7-Nachrichten von ORBIS, die von JAIF erzeugt wurden. Ab dem Service Pack 07.05 erfolgte für die Kommunikation von ORBIS nach SAP die Umstellung auf direkte BAPI-Aufrufe aus dem Schnittstellenmodul JAIF heraus, wie schon zuvor für Diagnosen, Prozeduren und einige weitere Daten. Die Kommunikation über direkte BAPI-Aufrufe aus JAIF erfolgt bidirektional: Bei Aufruf eines BAPIs zur Anlage einer Patientenbewegung in SAP wird als Rückgabeparameter die Bewegungsnummer in die ORBIS-Datenbank übernommen.

Nach den BAPI-Aufrufen entstehen in SAP ebenfalls entsprechende HCM-Nachrichtenereignisse. Hieraus ergibt sich für das an SAP gekoppelte ORBIS ein Echoeffekt. Ein von ORBIS nach SAP per HL7/BAPI kommunizierte Patientenbewegung wird über den Weg HCM/HL7 wieder als Echo an ORBIS zurückkommuniziert. Dieses Echo wird von JAIF verworfen. Subsysteme, die ADT-Daten aus ORBIS oder SAP benötigen, können jedoch aufgrund des Echomechanismus mit ADT-Daten aus beiden Systemen versorgt werden. Sie erhalten ihre HL7-Nachrichten ebenfalls aus dem transformierten HCM-Datenstrom. Bei gleichzeitiger Änderung von Daten in beiden Systemen, z.B. des Namens eines Patienten, setzt sich ORBIS durch.

Für Patienten- und Fallnummer wurden in beiden Systemen jeweils zwei Nummernkreisintervalle überschneidungsfrei konfiguriert, die von den beiden Systemen exklusiv bei einer Aufnahme verwendet werden.

ORBIS stellt bei der Dokumentation von Diagnosen und Prozeduren das führende System dar. Nach Kommunikation einer Aufnahmediagnose, eines Kostenträgers, der Aufnahmeart, der voraussichtlichen Verweildauer und des Einweisers nach SAP versendet der §301-Nachrichtenaufbereiter in SAP die Aufnahmeanzeige im EDIFACT-Format an die Krankenkasse.

Diagnosen und Prozeduren werden sofort nach Erfassung in ORBIS nach IS-H kommuniziert. Nach Abschluss der medizinischen Dokumentation wird der Fall in ORBIS zur Abrechnung freigegeben, die Freigabe sowie die Auszeichnung der Fachabteilungsentlass- und Krankenhaushauptdiagnose wird dann an IS-H kommuniziert.

3.3 Systemtechnik

Die Installation des Kommunikationservers e*Gate (Version 4.5.3) sowie des ORBIS-Schnittstellenmoduls JAIF erfolgte auf einem hochverfügbaren Cluster, siehe Abb. 1.

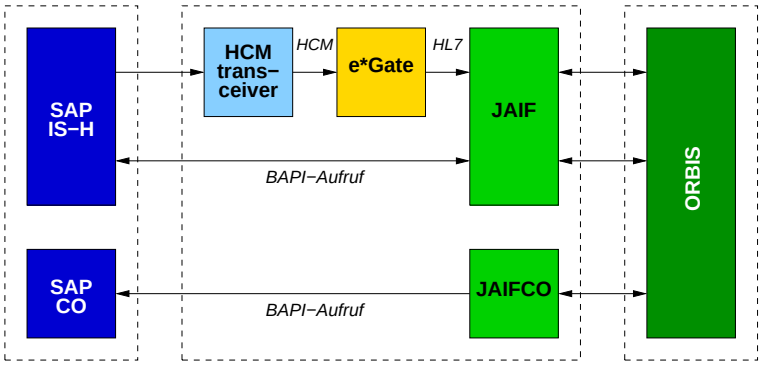

Abb. 1. Systemtechnische Realisierung der Koppelung: Auf einem hochverfügbaren Cluster sind der HCM-Transceiver, der als Client zum SAP-System die HCM-Daten exportiert, der Kommunikationsserver e*Gate zur Umwandlung von HCM nach HL7, sowie das ORBIS-Schnittstellenmodul JAIF installiert. Für die Anbindung von ORBIS an SAP CO existiert eine weitere JAIF-Instanz (geplant).

Die Schnittstelle kann mit dem Modul JAIFC überwacht werden. Hiermit kann insbesondere festgestellt werden, welche Nachrichten warum nicht vom jeweiligen Zielsystem verarbeitet werden konnten.

4 Vorbereitungen zum Produktivstart

4.1 Basisdatenpflege und Altdatenmigration

Neben der Basisdatenpflege in ORBIS war die Altdatenmigration ein wichtiger Meilenstein für die Inbetriebnahme. Zum Zeitpunkt des Produktivstarts des KAS sollte dieses mit dem gleichen Datenbestand wie das SAP-System starten, damit einerseits auch SAP-Altfälle in ORBIS bearbeitbar bleiben, andererseits eine Altdatenmigration aus klinischen Subsystemen möglich wird, die die SAP-Fallnummer als Ordnungskennzeichen führen.

Aus SAP wurden ab dem Zeitpunkt der Inbetriebnahme folgende Daten exportiert und in ORBIS importiert: Externe Ärzte und Krankenhäuser, Krankenkassen, Kostenträger, Patienten, Angehörige, Neugeborene, Fälle, stationäre Aufenthalte, ambulante Besuche, OP-bezogenen Prozeduren und Diagnosen, diese allerdings erst ab dem 01.01.2001.

Der Altdatenexport lässt sich innerhalb von 35 min bewerkstelligen, der Import in die ORBIS-Datenbank zog sich allerdings inklusive Korrekturen über fast 9 Tage hin. Die während dieser Zeit in SAP geänderten Bewegungsdaten wurden dann als HCM-Datei gesammelt, über den Kommunikationsserver in das HL7-Format konvertiert und über JAIF innerhalb weniger Stunden importiert. Die während der 9 Tage neu erfassten Diagnosen und Prozeduren wurden noch anschließend eingespielt, so dass zum Zeitpunkt der Inbetriebnahme nahezu ein Gleichstand des Datenbestandes vorlag. Die Fehlerquote für den Altdatenimport lag im Projekt zwischen 0 – 5%.

4.2 Integrationstests

Durch zwei umfangreiche Tests wurde die Funktion der Schnittstelle vor Produktivstart überprüft. Zunächst wurde ein ORBIS-Testsystem mit einem SAP-Qualitätssicherungssystem bidirektional gekoppelt. In ORBIS wurden zuvor die Daten des SAP-Qualitätssicherungssystem per Altdatenmigration übernommen. Mit einem standardisierten Prüfplan wurden nahezu alle Schnittstellenfunktionen geprüft.

Im Rahmen eines zweiten Tests wurde ein ORBIS-Testsystem mit ADT-Daten aus dem SAP-Produktivsystem beschickt, um das Verhalten der Schnittstelle unter Echtlastbedingungen zu überprüfen. Die Daten des Echtsystems wurden ebenfalls per Altdatenmigration übernommen.

5 Erfahrungen im Echtbetrieb

Bedingt durch den Pilotcharakter der Schnittstelle treten im täglichen Betrieb noch Fehler auf, die zeitaufwendige Korrekturen erfordern. Die Abrechenbarkeit von Leistungen in SAP ist aber grundsätzlich gewährleistet. Ohne ständige Unterstützung durch GWI ist die Schnittstelle derzeit noch nicht betreibbar. Selbst wenn jedoch die Schnittstelle einen höheren Reifegrad erlangt hat, ist eine fehlerfrei arbeitende Schnittstelle Illusion. Erst mit der Realisierung eines für das UKB ausreichenden Funktionsumfangs im Patientenmanagement- und Abrechnungssystem im KAS kann jedoch ein Umstieg auf die einfachere erste Schnittstellenvariante diskutiert werden.

Unterscheiden muss man grundsätzlich zwischen primären Schnittstellenfehlern und Folgefehlern. Bei den primären Schnittstellenfehlern lassen sich weiterhin zwei Gruppen unterscheiden,

1. Fehler die sich mit zunehmendem Reifegrad der Schnittstelle beseitigen lassen:
   - Fehler in Programmkomponenten und BAPIs.
   - Differenzen in den Basisdaten zwischen beiden Systemen. Beispiel: Die Kommunikation einer Verlegung scheitert, wenn das Zimmer im Zielsystem nicht vorhanden ist.
   - Fehler bei Kommunikationseventerzeugung.

- Fehler bei der Prüfung von Benutzereingaben. Beispiel: Im Quellsystem lässt sich eine Aufnahme mit einer unzulässigen Fachabteilungs-Stationskombination durchführen, die vom Zielsystem zurückgewiesen wird.
- Verwendung von Transaktionen im Quellsystem, die im Zielsystem nur unter bestimmten Umständen zulässig sind. Beispiel: Zusammenlegen von Patientenstammdatensätzen in SAP, wenn an beiden Datensätzen noch Fälle hängen. Dies ist in SAP möglich, in ORBIS hingegen nicht.

2. Grundsätzliche, durch die Schnittstellenkonzeption bedingte systemimmanente Fehler: Durch die zeitweise Nichtverfügbarkeit der Schnittstelle kommt es zur Doppelaufnahmen von Patienten und Fällen.
   Bevor die Rückkommunikation der ADT-Daten über direkte BAPI Aufrufe realisiert wurde, bestand in der Vergangenheit bei der Rückkommunikation über HL7-Nachrichten das Problem, dass HL7-Nachrichten von ORBIS verworfen wurden, wenn der Fall in SAP gerade zur Bearbeitung gesperrt war. Nach Umstellung auf die direkte BAPI-Kommunikation werden solche geblockten Nachrichten und die zu diesem Fall nachfolgenden Nachrichten, zunächst zurückgestellt, dann erneut kommuniziert bis die Kommunikation erfolgreich war.

Schnittstellenfehler wirken sich in der Regel so aus, dass die Freigabe der betroffenen Fälle zur Abrechnung fehlschlägt und die Fälle somit in SAP nicht abgerechnet werden können. Ca. 1% aller stationären Fälle sind mit abnehmender Tendenz davon betroffen und bedürfen der Nachkorrektur.

Das ORBIS-System am UKB wird von etwa 1.600 Benutzern flächendeckend in allen Kliniken genutzt. Obligat werden die DRG-relevanten Funktionen eingesetzt, optional u.a. die Arztbriefschreibung, die Konsilanforderung sowie die OP-Planung und -Dokumentation. Realisiert sind weitere Schnittstellen zur Übermittlung von Labordaten.

Danksagung Den zahlreichen Mitarbeiterinnen und Mitarbeitern der GWI und des UKB, die mit ihrem Einsatz zum Gelingen dieses Projektes beigetragen haben, sei an dieser Stelle gedankt.

Literatur

1. Kuhn, K., Giuse, D.: From hospital information systems to health information systems. Method. Inform. Med. **4** (2001) 275–287
2. Morzinck, T., Schneichel, W.: PDV-KAS-Schnittstelle: Ein Bericht aus der Praxis. In: GMDS/SGMI Tagung 2004. German Medical Science (2004)
3. Niemann, H., Hasselbring, W., Wendt, T., Winter, A., Meierhofer, M.: Kopplungsstrategien für Anwendungssysteme im Krankenhaus. Wirtschaftsinformatik **44** (2002) 425–434
4. SAP AG: IS-HCM Guide for External System Partners. (1999)
5. Heitmann, K., Blobel, B., Dudeck, J.: HL7-Kommunikationsstandard in der Medizin. 1 edn. Mönch Verlag (1999)
6. Herth, B., Laroque, S., Prinz, A.: SAP Basissystem. In CDI, ed.: SAP Anwenderedition. 1 edn. Addison-Wesley, Bonn (1998)

Integration von Anwendungssystemen des stationären und ambulanten Versorgungssektors am Beispiel des Projektes Mamma@kte.nrw

Peter Haas, Mandy Eckenbach, Hermann Plagge, Witold Schiprowski

Fachhochschule Dortmund, eHealth Presentation- and Evaluation-Center (EHPEC)
Emil-Figge-Str. 42, 44227 Dortmund

Zusammenfassung. Der Aufbau sektorübergreifender medizinischer Versorgungsdokumentationen erfordert die Integration von Medizinischen Informationssystemen aus dem ambulanten und stationären Bereich, die prinzipiell keinen gemeinsamen Kommunikations- und Dokumentationsstandard unterstützen. Es wird vor diesem Hintergrund der Aufbau der sektorübergreifenden krankheitsbezogenen Mamma@kte.nrw beschrieben. Dabei wird auf die technische, sicherheitstechnische Architektur und Aspekte der semantischen Integration des EAI-Projektes eingegangen sowie auf die Notwendigkeit von Vereinbarungen eines zentralen Dokumentationsschemas und von Standarddokumenten. Vor diesem Hintergrund wird die implementierte EAI-Architektur – die sowohl eine Punkt-zu-Punkt-Kommunikation unterstützt, als auch eine zentrale Krankenakte als Integrationselement beinhaltet – dargestellt. Die bisherigen Erfahrungen werden aufgezeigt.

1 Einführung

Die Integration heterogener Informationssysteme im Gesundheitswesen erfolgte seit etwa Anfang der 80er Jahre im Wesentlichen innerhalb von Krankenhäusern und dort unter Einsatz von speziellen Integrationsplattformen – den so genannten Kommunikationsservern [1] – durch Austausch von Nachrichten definierter Nachrichtentypen. Dabei standen datenschutzrechtliche Aspekte aufgrund des innerbetrieblichen Charakters dieser Kommunikationen im Hintergrund. Die Kopplungen waren meist zunehmend unaufwändig zu realisieren, da alle kommerziellen Produkte den HL7-Kommunkationsstandard [2] unterstützten. Im ambulanten Bereich wurden fast zeitgleich eine ganze Reihe von Austauschformaten definiert, die so genannten xDT-Standards [3]. Diese dienten zur Übertragung von Abrechnungsinformationen (ADT) und Laborergebnissen (LDT), aber auch zur Befundkommunikation oder aber, um bei Systemwechseln die Datenmigration zu unterstützen (BDT). Die parallelen Entwicklungen führten dazu, dass noch heute in den beiden Versorgungssektoren verschiedene Ausgangssituationen für sektorübergreifende EAI-Projekte vorliegen.

Vor dem politischen Hintergrund einer kritisierten Versorgung von Patientinnen mit Mammakarzinom sowie den gesetzlichen Möglichkeiten einer integrierten Versorgung entstand 2003 der politische Wille in Nordrhein-Westfalen, eine spezielle

sektorübergreifende elektronische Dokumentation für die Behandlung von Patientinnen mit einem Mammakarzinom zu realisieren. Dabei sollten folgende Randbedingungen berücksichtigt werden:

- Basis für die Elektronische Krankenakte sollte die sichere Server-Lösung der KV Nordrhein D2D (Doctor to Doctor) bzw. die technische Lösung PaDok der Fraunhofer Gesellschaft St. Ingbert sein [4].
- Die Einbindung der Praxissysteme sollte über den sicheren Kommunikationsstandard VCS erfolgen, da ein Großteil der bei den beteiligten Praxen eingesetzten Arztpraxisinformationssysteme das VCS-Verfahren unterstützte [5].
- Es sollte das zentrale Datenschema der KV Nordrhein für die Mammakarzinom-Dokumentation berücksichtigt werden [6].

In der Folge wurden zwei Pilotregionen in Essen und Düsseldorf definiert, wobei die Region Essen fünf niedergelassene Gynäkologen, drei Krankenhäuser und eine radiologische Praxis umfasst. Damit waren in der Pilotregion Essen insgesamt drei verschiedene Arztpraxisinformationssysteme, drei verschiedene Krankenhausinformationssysteme, ein Radiologieinformationssystem und ein Tumordokumentationssystem zu einem funktionierenden Ganzen zu integrieren sowie die Interoperabilität zwischen VCS und PaDok herzustellen.

2 Integrationsaspekte

2.1 Realisierte Systemarchitektur

Prinzipiell kann der Aufbau einer gemeinsamen Dokumentation bzw. ein Netz interoperierender Systeme mittels der zwei nachfolgend dargestellten Lösungsansätze realisiert werden, wobei komponenten- und dienstbasierte Ansätze – wie sie z.B. für das Gesundheitswesen bei CORBAmed beschrieben werden [7] – aufgrund des proprietären Charakters der im Projekt involvierten kommerziellen Systeme außer acht gelassen werden mußten:

1. Interoperation durch direkte Kommunikation („Kommunikationslösung“)
   Die Systeme kommunizieren direkt oder über einen zentralen Kommunikationsserver miteinander über den Austausch von Nachrichten, angehängten Dokumenten und auf Basis eines definierten Kommunikationsstandards.
2. Interoperation durch gemeinsamen persistenten Speicher („Speicherlösung“)
   Die Systeme speichern die für andere Institutionen wichtigen Informationen in einem zentralen Server oder auf einem mobilen Datenträger z.B. einer Chipkarte, von dort können sie dann von anderen berechtigten Teilnehmern abgerufen werden. Dies kann ebenfalls durch Versenden entsprechender Nachrichten an dieses Zielsystem oder aber durch Nutzung von Methoden dieses Zielsystems geschehen. Ein entsprechendes methodenbasiertes Verfahren für eine zentrale Basisdokumentation wird in [8] skizziert.

Im letztgenannten Fall kann man von einer *einrichtungsübergreifenden Elektronischen Patientenakte* sprechen, die auch – bei Einsatz einer zentralen Serverlösung – integriert Kommunikationsfunktionen übernehmen kann, in dem z.B. neu eingestellte Daten und Dokumente an bestimmte vom einstellenden System angegebene Empfänger übersandt werden. Beispielhafte Implementierungen eines solchen Lösungsansatzes sind die Produkte „PaDok“ (Fraunhofer Gesellschaft) [4] und „ehealthConnect“ (Fa. MEDNET AG.) [9].

Da aus praktischen und juristischen Gründen zentrale Akten in naher Zukunft die lokalen Dokumentationen in den einzelnen Einrichtungen nicht ersetzen werden, erfordern beide Lösungsansätze, dass institutionelle Informationssysteme über ein Kommunikationsmodul (oftmals als *Konnektor* [10] bezeichnet) verfügen, mittels dem sie mit anderen Systemen oder einer einrichtungsübergreifenden Akte kommunizieren und Dokumente austauschen können. Das Informationssystem sollte neben dem Konnektor auch über einen eMail-Client verfügen, mittels dem ein-/ausgehende Nachrichten protokolliert und von berechtigten Nutzern eingesehen werden können.

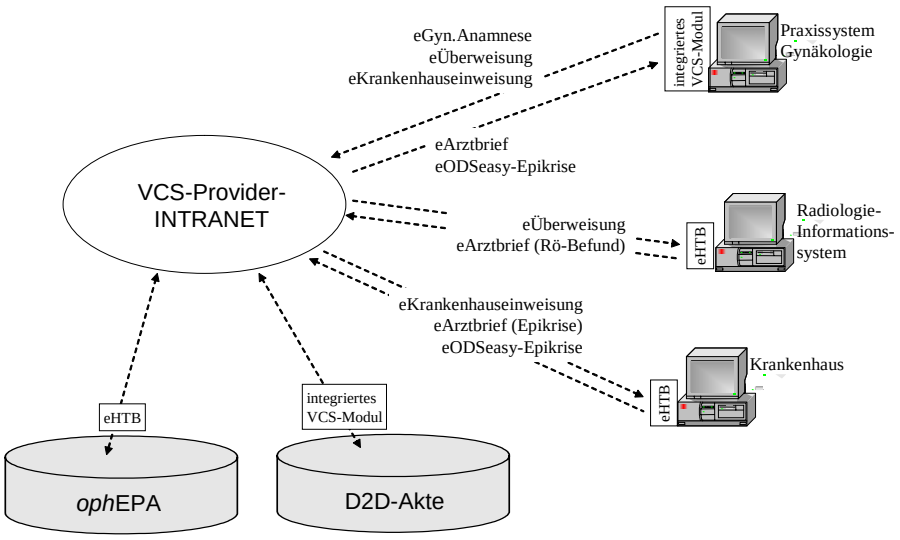

Abb. 1. Integrationsszenario Mamma@kte.nrw

2.2 Sicherheitsinfrastruktur

Das durch die KV Nordrhein und die Landesregierung NRW initiierte Projekt hat im Kern das Ziel, unter Nutzung der VCS-Sicherheitsinfrastruktur die zentrale Akte D2D zum Einsatz zu bringen. Es wird daher für die Kommunikation der VDAP Communication Standard (VCS) [5] verwendet, welcher derzeit SMTP, S/MIME als Transportprotokoll benutzt. Ein VCS-eigenes Kommunikationsprotokoll (oberhalb von SMTP) ermöglicht einen eigenen Quittungsbetrieb versendeter und empfangener Nachrichten. Der Transport wird dabei über eine vom Internet getrennte gesicherte Intranet-Verbindung der VCS-Provider abgewickelt. Dabei erfolgt eine Signierung und Ver-

schlüsselung der Nachrichten mittels spezieller von den VCS-Providern ausgegebener Signaturkarten. Ein Kommunikations- und Prozessmodul („KPM“) bildet die technische Basis für den Nachrichtenaustausch. Kommunikation im Provider-Intranet, Adressbuchanfragen, Signatur und Signatur-Prüfung, Verschlüsselung sowie Protokollierung zu den versendeten und empfangenen E-Mails werden durch dieses Modul realisiert. Als technisches Nachrichtenformat benutzt VCS eine eigene Struktur auf Basis von S/MIME, hinzu kommen zur Inhaltsdefinition – ebenfalls VCS-spezifische XML-Dateien für Steuerinformationen und Nachrichteninhalte. Diese Nachrichteninhalte gehorchen bestimmten Geschäftsvorfällen (eÜberweisung, eArztbrief, eKrankenhauseinweisung), es können dabei beliebige Dateien als Anhänge mit übermittelt werden.

Mit VCS stand also zu Projektbeginn eine sichere „Kommunikationslösung“ zur Verfügung, diese wird jedoch inzwischen nur noch von wenigen Arztpraxisinformationssystemen unterstützt. Vor diesem Hintergrund musste für den weiteren Projektfortschritt ein VCS-fähiges Kommunikationsmodul für jene Informationssysteme realisiert werden, die VCS nicht unterstützen. Es wurde daher ein „eHealth Telematic Broker“ (eHTB) entwickelt, der die Funktionalität eines eMail-Clients hat und die gesamte VCS-Kommunikation abwickelt. Der eHTB kann alternativ lose über einen isolierten Aufruf und isolierte Nutzung oder sehr eng durch Integration auf Datenbank- und Funktionsebene mit Informationssystemen gekoppelt werden. Im letzeren Fall stellt das den eHTB nutzende Informationssystem Steuerinformationen in die Datenbanktabellen des eHTB ab und ruft integriert in eigenen Anwendungsfunktionen Signaturfunktionen oder Sende-/Empfangsfunktionen des eHTB auf. Ebenso können die empfangenen Daten und Dokumente automatisch in das Informationssystem inkooperiert werden. Um ein zur Kommunikation zeitgleiches Einstellen von Dokumenten in eine zentrale Akte zu ermöglichen, wurde im eHTB die Option realisiert, automatisch oder durch Benutzersteuerung die übermittelten Dokumente auch an eine zentrale Krankenakte zu senden. Bei der Architektur des eHTB wurde berücksichtigt, dass dieser auch modular um andere Kommunikationsverfahren als VCS erweitert werden kann.

Als zentrale Akte fungiert das PaDok-System (Synonym: D2D-Akte), das im Wesentlichen das Einstellen von signierten und verschlüsselten Dokumenten erlaubt und auch eine nicht-adressierte („ungerichtete“) Kommunikation z.B. für Überweisungen unterstützt (zum Gesamtzusammenhang s. Abb. 1).

2.3 Semantische Geschäftsprozessintegration

Soll eine einrichtungsübergreifende Elektronische Krankenakte mehr als eine lose Sammlung von Dokumenten sein, wird die Verwaltung von Metadaten notwendig [11], gegebenenfalls soll auch die Dokumentation selbst in strukturierter und formalisierter Form vorliegen. Für Letzteres wurde von der KV Nordrhein zusammen mit einem Industriepartner ein XML-Schema für eine Mamma-Tumordokumentation zur Verfügung gestellt [6], welches klinisch-epidemiologischen Charakter hat. Aus diesem Schema wurden einzelne so genannte „*Schemadokumente*“ abgeleitet (siehe Abbildung 2), die Struktur und Semantik einzelner Dokumente definieren. Schnell zeigte sich jedoch, dass in diesem formalen epidemiologisch orientierten Schema die eher informale Versorgungsdokumentation der ambulanten Projektteilnehmer in den Arzt-

praxissystemen nicht berücksichtigt war und eine Überführung aller ambulanten Informationen in strukturierte und formalisierte Dokumente mit den entsprechenden Funktionen nicht sach- und kostengerecht sein konnte. Dementsprechend wurde für eine erweiterte Kommunikation und Dokumentation auf Basis der Clinical Document Architecture (CDA) [12] eine erweiterte Überweisung und ein Arztbrief definiert – im Projekt als „*Versorgungsdokumente*“ bezeichnet. Schema- und Versorgungsdokumente bilden in ihrer Gesamtheit die Mamma@kte.nrw.

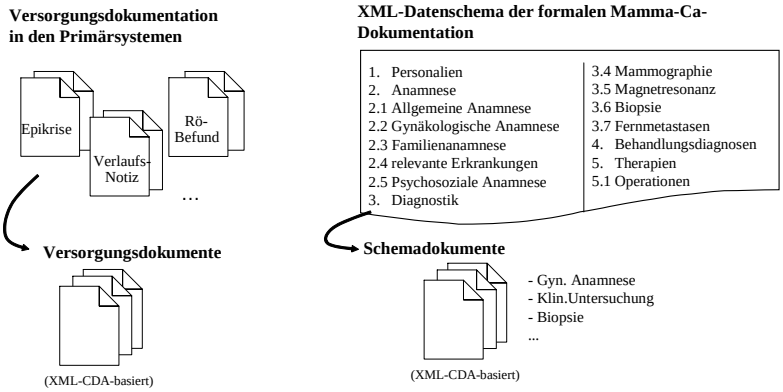

Abb. 2. Datenobjekte der Mamma@kte.nrw: Versorgungs- und Schemadokumente

Als weitere Basis des Projektes wurde zu Beginn ein typischer Behandlungsprozess bestehend aus über 80 Einzelmaßnahmen definiert und die entsprechenden Dokumente zugewiesen (siehe Beispiel Abbildung 3). Anhand dieses Prozessverlaufes wurde auch festgelegt, welche Versorgungsinstitution wann Dokumente an wen übermittelt bzw. in die Akte einstellt.

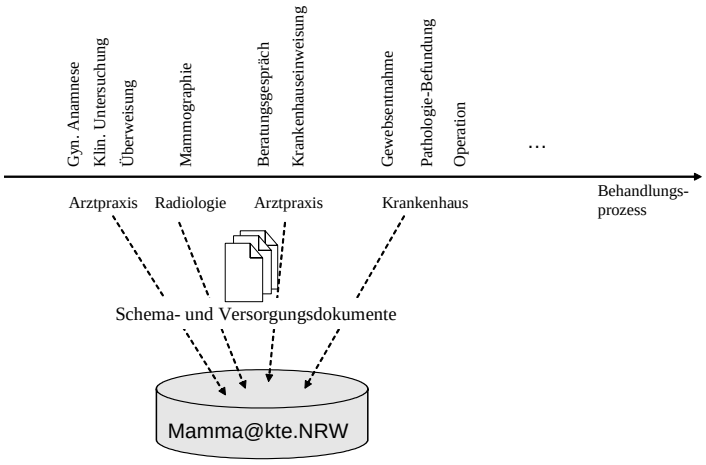

Abb. 3. Behandlungsprozess, Dokumente und zentrale Akte

Da PaDok im Wesentlichen nur das Einstellen von verschlüsselten Dokumenten in den zentralen Server erlaubt, aber keine domänenspezifischen Metadaten unterstützt, wurde zu Evaluations- und Forschungszwecken parallel ein CDA-Level1-kompatibles

Schema für eine zentrale Akte entworfen und datenbanktechnisch implementiert. Diese ontologie-, prozess- und handlungsorientierte Elektronische Patientenakte (o*ph*EPA) ist Teil des eHealth Presentation- and Evaluation-Centers (EHPEC), ihre Integration in das Netz wurde ebenfalls mittels des HTB vorgenommen.

Insgesamt ergibt sich vor diesem Hintergrund das in Abbildung 1 gezeigte EAI-Szenario. Mittels dieser Infrastruktur können die Projektteilnehmer einerseits gezielt und ohne die Dokumente in eine zentrale Akte einzustellen sicher sektorübergreifend die Versorgungsdokumente oder anderen Schriftwechsel und Dokumente untereinander kommunizieren, andererseits ist es aber auch möglich, alle oder ausgewählte Behandlungsdokumente in die D2D-Akte oder die o*ph*EPA einzustellen.

Eine einrichtungsübergreifende Elektronische Patientenakte als Integrationsinstrument zeichnet sich gegenüber der Kommunikationslösung durch die persistente Verfügbarkeit aller Behandlungsdokumente für die Teilnehmer aus. Zu jeder Zeit kann ein Rückgriff auf die in der zentralen Akte hinterlegten Informationen und das Einstellen eigener Behandlungsdaten erfolgen. Allerdings hängt der Zugriff auf die Akte bzw. auf Teile davon von der Einwilligung des Patienten ab. Typischerweise wird diese Einwilligungserklärung technisch mithilfe eines temporär gültigen Zugriffsschlüssels – meist gespeichert auf einem mobilen Datenträger z.B. einer Chipkarte – umgesetzt. Überreicht der Patient diesen einem behandelnden Arzt, wird dieser damit berechtigt, auf die damit assoziierten und zentral hinterlegten Behandlungsdaten (temporär) zuzugreifen. Der für den einzelnen Arzt zugreifbare Informationsumfang einer konkreten Elektronischen Patientenakte kann dabei die Vollversorgung, die indikationsbezogene Versorgung sowie die Einzelkooperation oder nur ausgewählte Notfallinformationen umfassen.

3 Projektvorgehen und Projektstatus

Das Projekt befindet sich noch in der stufenweisen Umsetzung. Die beschriebenen Integrationsszenarien wurden bereits im „eHealth Presentation- and Evaluation-Center“ implementiert, im Routineeinsatz ist zurzeit die Kommunikationslösung.

In der aktuell laufenden ersten Stufe kommunizieren die Leistungserbringer Versorgungsdokumente und Schema-Dokumente der Mammakarzinom-Dokumentation gerichtet mittels dem VCS-Standard. Ein beispielhaftes Kommunikationsszenario sieht wie folgt aus: Eine gynäkologische Arztpraxis übermittelt per VCS über einen VCS-Provider (z.B. DGN, Telemed) die gynäkologische Anamnese als CDA-Dokument an ein Krankenhaus. Das Krankenhaus nimmt die VCS-Nachricht entgegen und integriert die Angaben im Tumordokumentationssystem „ODSeasy“ oder direkt im krankenhauseigenen Informationssystem (KIS). Nach der Behandlung der Patientin im Krankenhaus wird die freitextliche Epikrise („Arztbrief“) sowie die aus der formalen Tumordokumentation des Systems ODSeasy exportierte ODSeasy-Epikrise elektronisch an den einweisenden Arzt per VCS versandt, der diese dann teilautomatisiert in die entsprechende Karteikarte in seinem Praxisinformationssystem ablegt.

In der nun anstehenden zweiten Stufe soll die ungerichtete Kommunikation zwischen den Systemen über den VCS-Standard mit einbezogen werden. Diese erweiterte Kommunikationsmöglichkeit lässt folgendes Kommunikationsszenario zu: Die Pati-

entin erhält vom einweisenden Arzt eine ausschließlich diesem Vorgang zugewiesene TAN für die Weiterhandlung im Krankenhaus, wobei die damit einhergehende generierte eKrankenhauseinweisung unter Hoheit des Absenders in einen Zwischenspeicher transferiert wird. Der weiterbehandelnde Arzt im Krankenhaus kann über diese von der Patientin übergebene TAN eine Anforderung an den Absender senden, durch die die Identität und Berechtigung des Empfängers geprüft werden kann. Bei erfolgreicher Prüfung wird die Überweisung automatisch verschlüsselt und an das Krankenhaus gesandt.

In der dritten Stufe wird der zentrale Aktenserver PaDok oder eine alternative Lösung integriert. Fallbezogen werden dann Daten von den Behandlern in der Akte bereitgestellt. Die VCS-/D2D-Interoperabilität wurde von den Projektpartnern bereits realisiert und steht im EHPEC zur Verfügung.

4 Erfahrungen und Diskussion

Der Aufbau einer elektronischen Kommunikation zwischen Informationssystemen des ambulanten und stationären Versorgungssektors sowie die Etablierung einer einrichtungsübergreifenden Krankenakte als Integrationsplattform ist heute immer noch kein Plug-and-Play-Projekt. Unterschiedliche Kommunikationsstandards und unterschiedliche oder fehlende Schnittstellen der Informationssysteme sowie grundsätzlich verschiedene Dokumentationsparadigmen erschweren eine Interoperabilität erheblich. Folgende wesentliche Erfahrungen können festgehalten werden:

- Ärzte in Praxen und Krankenhäusern sind prinzipiell motiviert und engagiert hinsichtlich des Aufbaus einer gemeinsamen elektronischen Kooperationsplattform.
- Die Ausarbeitung beispielhafter Prozessszenarien und Krankenakten ist ein hilfreicher Einstieg in sektroübergreifende Integrationsprojekte.
- Durch den VCS-Standard kann eine sichere und vertrauenswürdige Kommunikationsinfrastruktur aufgebaut werden, wobei Signieren und Verschlüsseln von Dokumenten mit fortgeschrittenen Signaturen einen erheblich höheren Zeitaufwand erfordern, als das konventionelle Unterschreiben von Papierdokumenten.
- Eine Asynchronität zwischen konventionellen und elektronischen Prozessen erschwert zeitnahes und sachgerechtes elektronisches Signieren und Versenden von Versorgungsdokumenten – vor allem bei Informationssystemen, die kein elektronisches „Freigeben“ keinen Workflow von Dokumenten unterstützen.
- Hersteller kommerzieller Systeme haben nur bedingt Interesse, sich in generischer Weise für eine Interoperabilität mit anderen Informationssystemen zu öffnen.
- Die Integration von XML-Dokumenten in Praxisinformationssysteme scheint aufwändig. Alleine die Realisierung der gynäkologischen Anamnese dauerte Monate und wurde von Herstellern als sehr umfangreich bezeichnet.
- Die Krankenhäuser haben erhebliche Sicherheitsbedenken und Ängste in Bezug auf eine Vernetzung mit Externen und Krankenhausinformationssysteme sind für eine Vernetzung mit dem ambulanten Sektor nur bedingt ausgelegt. Der Aufwand zur Herstellung einer sicheren Plattform ist erheblich.

- Für sektorübergreifende Integrationsprojekte ist es notwendig, über einen herstellerunabhängigen Software-Konnektor (im Projekt: eHealth Telematic Broker) zu verfügen, mittels dem Informationssysteme in ein elektronisches Versorgungsnetz integriert werden können.
- Die Clinical Document Architecture ist eine hervorragende Grundlage für die Kommunikation und zentrale Dokumentation, da integriert Metadaten und Inhalte dargestellt und kommuniziert werden können.
- Die Nutzung einer zentralen Akte, die kaum Metadaten beinhaltet und alle Informationen ausschließlich verschlüsselt speichert, führt zu aufwändigen Bedienungsaktionen für den Arzt in seinem institutionellen Informationssystem, was erheblich akzeptanzmindernd wirken kann.
- Eine nachhaltige Weiterführung des Projektes hängt entscheidend von der Loslösung von proprietären Standards ab. In diesem Sinne soll der eHTB zu einem transparenten offenen „Konnektor“ weiterentwickelt werden, der auch mit der erwarteten nationalen Telematikplattform für die Elektronische Gesundheitskarte interoperieren kann.
- Der spezielle Aspekt der Ausbalancierung von sicherheits- und dateschutztechnischen Lösungen und Aufgabenangemessenheit für die Benutzer soll eingehender untersucht werden.

Literatur

1. Heitmann K.U.: The Role of Communication Servers in the Architectur of Healthcare Information Systems. In: Dudeck et. al. (eds.): New Technologies in Hospital Informations Systems, IOS Press Amsterdam (1997) 156-162
2. HL7 Standard Version 2.3. Ann Arbor, Michigan: Health Level Seven 1997
3. http://www.zi-berlin.de/service/It/it/index.html, letzer zugriff 10.03.2005
4. Bresser,B., Paul,V.: "PaDok: Eine Lösung für Ärztenetze und Krankenhäuser der Grund- und Regelversorgung. Sichere Kommunikation und Kontrollstatistik für integrierte Versorgung". In Eissing U. (Hrsg.): MEDNET Arbeitsbuch für die integrierte Gesundheitsversorgung 2000/1, Edition Temmen Bremen (2000) 113-135
5. PAS 1011, Ausgabe: 2001-03 VCS Kommunikationskonzepte für das Gesundheitswesen, Beuth Verlag Berlin 2001
6. http://www.kvno.de/importiert/mammaakte.pdf, letzer Zugriff 10.03.2005
7. www.omg.org, letzer Zugriff 10.03.2005
8. Haas P.: Basisdokumentation und Elektronische Gesundheitskarte (Teil 2). In: Forum der Medizin_Dokumentation und Medizin_Informatik Heft 4 (2004) 150 - 153
9. Eissing U.: eHealthConnect und MEDNET Server. in: Eissing U. (Hrsg.): MEDNET Arbeitsbuch für die integrierte Gesundheitsversorgung 2002/3, Edition Temmen Bremen (2002) 101-111
10. IBM Deutschland GmbH et. al.: Erarbeitung einer Strategie zur Einführung der Gesundheitskarte – telematikrahmenarchitektur für das Gesundheitswesen, Eigenverlag 2004
11. Haas P.: Medizinische Informationssysteme und Elektronische Krankenakten, Springer Berlin Heidelberg 2005
12. Dolin, R.H et. al.: HL7 Document Patient Record Architecture: An XML Document Architecture Based on a Shared Information Model. In: Proc. AMIA Symp.: 52–56 (1999)

Potential der CDA in verteilten Gesundheitsinformationssystemen

Mario Beyer[1], Klaus A. Kuhn[2] und Richard Lenz[1]

[1] Philipps-Universität Marburg, Institut für Medizinische Informatik,
Bunsenstr. 3, D-35033 Marburg.
{beyerm|lenzr}@med.uni-marburg.de
[2] Technische Universität München, Institut für Med. Statistik und Epidemiologie,
Ismaninger Str. 22, D-81675 München.
klaus.kuhn@lrz.tum.de

Zusammenfassung. Die IT-Unterstützung institutionsübergreifender Behandlungsprozesse wird immer wichtiger im Gesundheitswesen. Für eine zukunftssichere Lösung stehen Erweiterbarkeit und Integration in bestehende Systeme im Mittelpunkt, und damit auch die Wiederverwendbarkeit der gespeicherten Informationen. Mit CDA („Clinical Document Architecture") steht eine flexible Dokumentenarchitektur zur Verfügung, die diese Ansprüche adressiert. Dieser Artikel befasst sich damit, inwiefern CDA über eine Nutzung als Dokumentationsformat hinaus als Informationsbasis in einem prozessorientierten Informationssystem im Versorgungsnetz eingesetzt werden kann.

1 Herausforderung Versorgungsnetz

Die Informationstechnik revolutioniert viele Bereiche des täglichen Lebens. Auch im Gesundheitswesen erhofft man sich ein hohes Verbesserungspotential durch Unterstützung der Arbeitsprozesse. Wie zahlreiche Studien belegen treten in der Medizin häufig Fehler auf, die zu einem großen Teil vermeidbar wären [1]. Als Ursachen lassen sich mangelnde Verfügbarkeit von Informationen, schlechte Kommunikation und Koordination, und ebenso die Komplexität dieser Anwendungsdomäne ausmachen [2, 3].

Vielfältige unterstützende IT-Anwendungen sind denkbar, von Wissensquellen zum schnellen, kontextbezogenen Nachschlagen über institutionsübergreifenden Datenaustausch und Telekommunikation hin zu entscheidungsunterstützenden Anwendungen. Aktuelle Entwicklungen im Gesundheitswesen wie der Ausbau der Integrierten Versorgung (sektorenübergreifende Behandlungskoordination), der Aufbau Medizinischer Versorgungszentren (spezialisierte ambulante Kliniken) und die Einführung von Disease-Management-Programmen (strukturierte Behandlungsprogramme zur Optimierung standardisierbarer Versorgungsabläufe für definierte Patientengruppen) zeigen den zunehmend verteilten Charakter der Versorgung, den ein IT-System reflektieren sollte. Leitlinien (evidenzgesicherte, systematische Entscheidungshilfen für bestimmte Krankheitsbilder) und Behandlungspfade (lokale, leitlinienbasierte Ablaufvorgaben für die Behandlung

bestimmter Fälle) versuchen Prozessvarianzen zu begrenzen und aktuelle Erkenntnisse in die Behandlung einfließen zu lassen. (vgl. auch [4])

Kernanforderungen an ein institutionsübergreifendes Informationssystem sind die *Integration* heterogener Systeme, so dass Informationen semantisch kompatibel ausgetauscht werden können, *Flexibilität* zur Anpassung des Systems auf regionale oder lokale Eigenheiten und *Evolutionsfähigkeit*, damit das System auf zukünftige Anforderungen reagieren kann, die im Laufe der Zeit entstehen, auch während der und durch die Systemeinführung [5]. Hierzu wäre es sinnvoll, eine IT-Infrastrukturplattform aufzubauen, auf der dann flexible Anwendungen je nach Bedarf aufsetzen können. Damit die Informationstechnik eine möglichst gute Unterstützung bietet, aber die realen Abläufe wenigstmöglich einschränkt, sollte sie sich an diesen orientieren, d. h. den Behandlungsprozess begleiten, wodurch der Patient als zentraler Bezugspunkt in den Fokus rückt.

Die Kernaufgabe der IT in einem Informationssystem lässt sich damit beschreiben als das Sammeln, Pflegen und vor allem Verknüpfen von patientenunabhängigem *Wissen* und patientenabhängiger *Information*. Dabei bilden eine patientenbezogene und institutionsunabhängige *elektronische Patientenakte* einerseits und der *Behandlungsprozess* andererseits die statischen und dynamischen Referenzobjekte.

In der vorliegenden Arbeit wird der Frage nachgegangen, wie eine institutionsunabhängige Patientenakte und eine geeignete Prozessbegleitung in einem verteilten Informationssystem umgesetzt werden können. In der Realität haben sich *Dokumente* in Papierform wie Arztbriefe und Rezepte etabliert. Diese sollen stabile, in einem gewissen Kontext gültige Informationen (Messwerte, Befunde etc.) zusammen mit ihrem Kontext festhalten und einen weitgehend standardisierten Informationsaustausch ermöglichen. Im Gegensatz zu diesen stabilen Daten stehen die volatilen operativen Daten (z. B. aktueller Aufenthaltsort), bei denen meist nur der aktuelle Wert von Interesse ist, und die meist in veränderlichen Datenbankfeldern abgelegt sind. Ein Ansatz basierend auf einer zentralen Informationsspeicherung etwa in einer relationalen Datenbank empfiehlt sich jedoch für eine institutionsübergreifende IT-Unterstützung nicht: Vorhandene semantisch heterogene Systeme sind nicht ohne weiteres integrierbar [6], die Festschreibung der Datensemantik in einem Datenbankschema ist zu unflexibel, außerdem muss ein Betrieb der Teilsysteme auch autonom ohne Online-Zugriff auf eine zentrale Datenbank möglich sein.

Ein rein dokumentbasierter Ansatz dagegen hat den Nachteil, dass operative Daten vielfach redundant in den Dokumenten vorgehalten und auch nicht aktualisiert werden, da dies der Philosophie der Dokumentorientierung widersprechen und immensen Aufwand bedeuten würde. Dennoch ist die Verwendung eines standardisierten Dokumentformats als systemunabhängige Einheit des Informationsaustauschs sinnvoll, da auf dieser Basis eine Entkopplung zwischen Informationsobjekten und den autonomen informationsverarbeitenden Systemen möglich wird.

Die Clinical Document Architecture (CDA, [7]) wird als Teil des Standards HL7 entwickelt und soll die informationstechnische Umsetzung austauschbarer

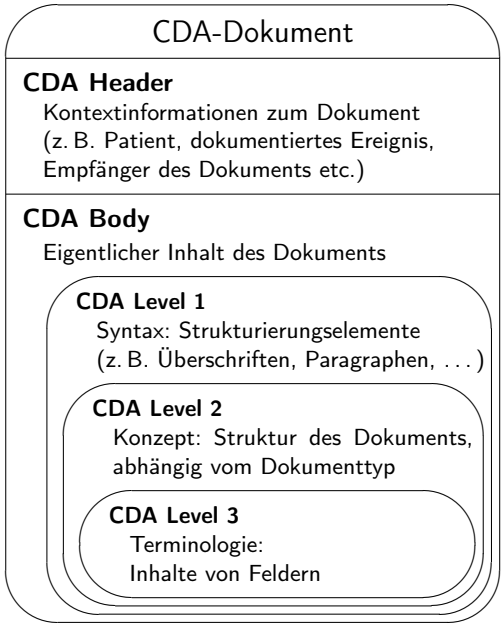

Abb. 1. Semantische Abstraktion bei CDA-Dokumenten

medizinischer Dokumente ermöglichen. CDA definiert basierend auf XML klinische Dokumente, wobei drei inhaltliche Abstraktionsstufen unterschieden werden (vgl. Abb. 1). Ebene 1 erfasst die Kontextdaten eines Dokuments, die Ebenen 2 und 3 sind für eine weitergehende semantische Annotation der Dokumentinhalte zuständig. So können einerseits auch unstrukturierte Dokumente zugeordnet werden, andererseits können strukturierte CDA-Dokumente zumindest angezeigt und vom Betrachter interpretiert werden, selbst wenn das System zur Interpretation nicht in der Lage ist. Auf diese Weise wird es ermöglicht, die semantische Kompatibilität der Primärsysteme auf der Basis einer gleichbleibenden Dokumentenarchitektur schrittweise zu verbessern.

Es stellt sich die Frage, inwieweit sich CDA über den reinen Dokumentationscharakter hinaus als Basis eines verteilten Gesundheitsinformationssystems nutzen ließe, so dass man von der Plattformunabhängigkeit und Evolutionsfähigkeit des CDA-Konzeptes profitieren könnte.

2 CDA als Basis für verteilte Patientenakte

Das CDA-Format enthält bereits auf Ebene 1 notwendige Kontextdaten zum Dokumentinhalt. Darunter fallen auslösendes Ereignis, Bezugspersonen (z. B. Patient, Arzt) oder maßnahmenbezogene Angaben (Maßnahme, Grund, Ort). Weiter werden Querbezüge hergestellt, bspw. kann zu einem Ergebnisdokument ein Anforderungsdokument über dessen ID referenziert werden oder es können

Beziehungen zu vorherigen Versionen desselben Dokuments („ersetzt“, „ergänzt“, ...) hergestellt werden. Damit ist also eine grundlegende Metadatenstruktur vorhanden, die für eine elektronische Patientenakte genutzt werden kann. Mit der zunehmenden Standardisierung von CDA wird auch die Detailliertheit medizinischer Spezifikationen steigen und ihr Normierungsgrad zunehmen, was die strukturelle und semantische Kompatibilität erweitern würde.

CDA erscheint damit interessant als Basis für eine verteilte Patientenakte. Im Zuge der Verbreitung von HL7 Version 3 kann auch davon ausgegangen werden, dass zunehmend Systeme in der Lage sein werden, CDA-Dokumente zu liefern oder weiterzuverarbeiten. Auch Haas [8] hält CDA als Datenbasis zwar für denkbar, sieht aber eine Nutzung der Daten für nachgeordnete Verwendung als erschwert oder unmöglich und schlägt vor, CDA für die Dokumentation von Ergebnissen als Spiegelbilder einer konventionell datenbankgestützten Akte zu verwenden. Dies brächte allerdings wieder eine im Datenbankschema festgeschriebene und damit unflexible Semantik mit sich.

In vielen Projekten zur elektronischen Patientenakte beschränkt sich diese auf eine gegliederte Sammlung von Dokumenten. Um das Nutzenpotiential auszuschöpfen, sollte sie aber deutlich darüber hinausgehen. Enthaltene Informationen müssen frei zugreifbar und auswertbar sein und sich mit im System abgelegtem Wissen verknüpfen lassen. Daher schlagen wir den in Abbildung 2 gezeigten Ansatz vor. Im Mittelpunkt steht dabei die Dokumentation mittels CDA-Dokumenten, die in einer geeigneten, bspw. verteilten Datenbank abgelegt werden (s. u.). Analog zum EAV-Ansatz (Entity–Attribute–Value, vgl. [9]), der eine generische Datenspeicherung in einer einzigen dreispaltigen Datenbanktabelle vorsieht, wird hier die Semantik nicht in einem Datenbankschema gespeichert, sondern bleibt offen bzw. wird dokumentspezifisch festgelegt. Semantikspezifikationen (z. B. Struktur der CDA-Dokumente, Domänenkonzepte, Terminologien) können ebenfalls in XML-Formaten – und damit in der Datenbank – abgelegt und mit XML-Mitteln verarbeitet werden (bspw. durch XSL-Transformationen), um die semantische Kontrolle der Inhalte zu ermöglichen. Mit Hilfe eines geeigneten Werkzeugs kann die Wartung der Spezifikationen nahe an den Endnutzer, also an Domänenexperten gebracht werden, ohne gleichzeitig IT-Fachwissen vorauszusetzen. Die Anbindung vorhandener Systeme wird durch den graduellen Aufbau der CDA ermöglicht, Fremddokumente können bei Vorhandensein minimaler Kontextinformationen zumindest zu einem CDA-Level-One-Dokument konvertiert („verpackt“) und damit korrekt zugeordnet werden.

An die Datenbasis wird nur die Anforderung gestellt, in einem Netzverbund XML-Daten verarbeiten zu können. Naturgemäß bietet sich hierfür eine XML-Datenbank an, für den hier genannten Ansatz wird zunächst lediglich eine beliebige verteilte Datenhaltung für XML-Dokumente vorausgesetzt, aus der durch eine transparente Zugriffsschicht Informationen abgefragt werden können. – Die Performanz von XML-Datenbanken bei komplexen Anfragen ist gegenüber konventionellen Datenbanken noch ein Nachteil, die Entwicklung schreitet jedoch voran. Um eine schnelle Verarbeitung auch großer Datenmengen für Auswertungszwecke zu ermöglichen, kann der Datenexport an ein Data-Warehouse-

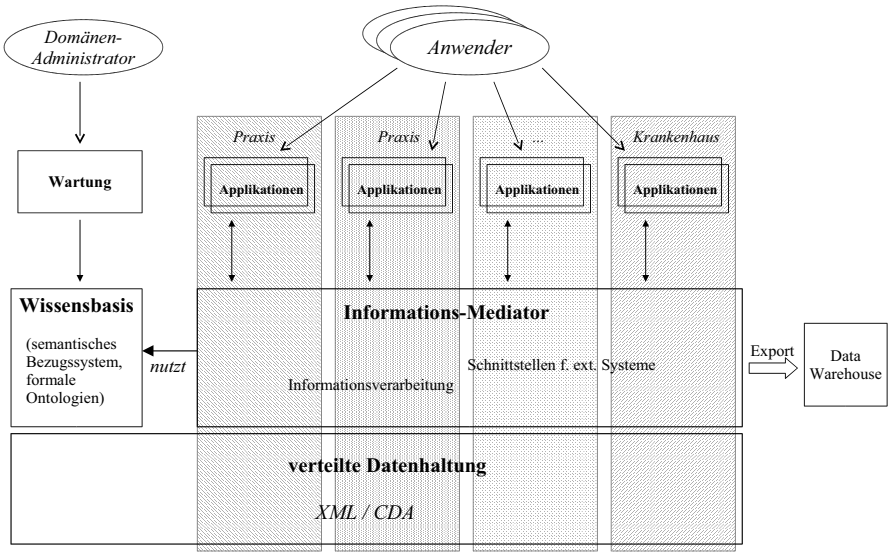

Abb. 2. Skizze einer CDA-basierten Informationssystem-Infrastruktur

System angebracht sein. Eine solche unidirektionale Anbindung schränkt die Flexibilität des beschriebenen Ansatzes nicht ein. Der Export kann bei Vorliegen von Semantikinformationen in der XML-Datenbank einfach durch Transformationen erledigt werden und ist ebenso flexibel konfigurierbar.

3 Prozessbegleitung

Der beschriebene Ansatz soll eine ausbaufähige, standardisierte Informationsbasis für die IT-Unterstützung im Versorgungsnetz ermöglichen. Dabei wurde bisher allerdings noch nicht auf eine Prozessorientierung eines solchen Informationssystems eingegangen. In der Praxis wird ein Prozess nicht zentral überwacht, stattdessen wird ein Patient von einem Laufzettel begleitet, der die notwendigen Informationen zu Prozess und Status enthält. Lassen sich auch CDA-Dokumente in dieser Funktion nutzen?

Eine Möglichkeit wäre die Einführung einer Art „Prozesszustandsdokument“ als Bezugsobjekt in einem verteilten Behandlungsprozess, dessen Semantik in Form eines XML-Dokuments in der Datenbank abgelegt werden könnte. Damit wäre auch die Prozesssemantik nicht festgeschrieben. Dazu müsste allerdings die Möglichkeit, Dokumente durch neue Versionen zu ersetzen, zweckentfremdet werden, um mit den eigentlich zustandslosen Dokumenten Zustände auszudrücken. Außerdem erlaubt CDA nur Rückwärtsreferenzen („ersetzt“-Verweise), da für Vorwärtsreferenzen Dokumente nachträglich geändert werden müssten. Weiter würden entweder ein ganzer Prozess durch ein einzelnes Dokument repräsentiert und damit verschiedene logische Dokumenttypen verknüpft, oder es

wäre zusätzlich zu den einzelnen Dokumenten in einem Prozess ein übergeordnetes Dokument zu übertragen, zu pflegen und aktuell zu halten, mit dem die beteiligten Systeme auch umgehen können müssten.

Die bessere Alternative wäre es, die Prozessinformation über die einzelnen Dokumente zu verteilen, so dass sich der jeweils nächste Prozessschritt ableiten lässt. Dazu müsste ein CDA-Dokument zumindest um eine Prozessstatusinformation erweitert werden, die den im Kontext des Dokuments gültigen (bzw. durch die im Dokument beschriebene Aktion erzielten) Status wiedergibt. Damit würde sich ein Prozess aus seinen Teildokumenten automatisch zusammensetzen lassen, sofern die Systeme die Information verstehen. Andere Systeme können das CDA-Dokument normal bearbeiten und die Information ignorieren, der Ansatz ist also kompatibel zur CDA-Philosophie. Die Semantik der Prozessinformationen kann analog zur Dokumentsemantik in Abschnitt 2 flexibel konfiguriert und inferiert werden. Benötigt würde noch eine Terminologie zur Prozessbeschreibung, die für die Standardisierung der Prozessinformation sorgt.

Mit einem entsprechenden Konfigurationswerkzeug kann die Spezifikation Domänenexperten ohne technische Datenbankkenntnisse übertragen und damit näher an den Benutzer gebracht werden. In einem verteilten Informationssystem ist der Konsens der beteiligten Parteien essentiell [10]. Es muss Einigkeit über gemeinsam benutzte Daten und Mittel sowie Abläufe bestehen. Das Konfigurationswerkzeug sollte daher einerseits als universelles Kontrollinstrument für Inhalte und Prozesse dienen und andererseits den Konsensfindungsprozess unterstützen. Bei der Konfiguration soll es dem Benutzer Konfigurationselemente anbieten, die den im System hinterlegten Konzeptbeschreibungen entsprechen (z. B. Datenobjekte aus CDA-Dokumenten) und dadurch dafür sorgen, dass systembekannte Konzepte eingesetzt werden, um eine Wiederverwendbarkeit der Daten sicherzustellen.

4 Zusammenfassung

In diesem Artikel wurde ein Ansatz für eine elektronische Patientenakte in einem Gesundheitsinformationssystem für ein Versorgungsnetz vorgestellt, in dessen Mittelpunkt CDA-Dokumente als zentrale Informationsobjekte stehen. Die in jedem CDA-Dokument vorhandenen Metainformationen ermöglichen die Zusammenführung zur Patientenakte. Darüber hinaus sorgen der graduelle Aufbau in mehrere semantische Abstraktionsschichten („Layer“) und die Möglichkeit der Semantikspezifikation für eine hohe Flexibilität und Evolutionsfestigkeit. Eine endnutzernahe Konfiguration mit geeigneten Werkzeugen erlaubt eine schnelle Anpassung durch medizinische Fachexperten. Weiter wird die Möglichkeit einer Prozessspezifikation durch geeignete Erweiterung von CDA-Dokumenten um Statusinformationen vorgeschlagen. Komplexe Auswertungen über größeren Datenmengen können mittels Datenexport in ein Data-Warehouse-System realisiert werden.

Entscheidend an diesem Ansatz ist, dass weder Semantikinformationen zu den enthaltenen medizinischen Daten noch Prozessinformationen fest codiert

werden, sondern dass sämtliche Konzepte selbst ebenso wie die eigentlichen Nutzdaten als Instanzen der Datenbank abgelegt sind. Diese Struktur bildet die Basis für verschiedenste Anwendungen und erlaubt die Verknüpfung von Informationen und Wissen sowie die evolutionäre Weiterentwicklung des Systems. Wichtig für eine Umsetzung ist die Verfügbarkeit performanter XML-Datenbanken mit weitreichenden Zugriffsstrukturen. XML stellt in diesem Zusammenhang nicht die Lösung des Problems „verteilte Akte“ dar, sie lässt sich jedoch mit geeigneten Konzepten wie der CDA so einsetzen, dass sie eine späte semantische Festlegung und damit die nötige Flexibilität für ein zukunftsfähiges System ermöglicht.

Literatur

1. Bates, D.W., Gawande, A.A.: Improving safety with information technology. N Engl J Med **348** (2003) 2526–2534
2. Kohn, L.T.: To Err Is Human. Building a Safer Health System. National Academy Press, Washington D.C. (2000)
3. Leape, L.L.: A systems analysis approach to medical error. J Eval Clin Pract **3** (1997) 213–222
4. Lenz, R., Beyer, M., Meiler, C., Jablonski, S., Kuhn, K.A.: Informationsintegration in Gesundheitsversorgungsnetzen – Herausforderungen an die Informatik. Informatik Spektrum **28** (2005) 105–119
5. Berg, M., Toussaint, P.: The mantra of modeling and the forgotten powers of paper: A sociotechnical view on the development of process-oriented ICT in health care. Int J Med Inf **69** (2003) 223–234
6. Colomb, R.M.: Impact of semantic heterogeneity on federating databases. The Computer Journal **40** (1997) 235–244
7. Dolin, R.H., Alschuler, L., Beebe, C., Biron, P.V., Boyer, S.L., Essin, D., Kimber, E., Lincoln, T., Mattison, J.E.: The HL7 Clinical Document Architecture. J Am Med Inform Assoc **8** (2001) 552–569
8. Haas, P.: Medizinische Informationssysteme und elektronische Krankenakten. Springer, Berlin, Heidelberg, New York (2005)
9. Nadkarni, P.M., Marenco, L., Chen, R., Skoufos, E., Shepherd, G., Miller, P.: Organization of heterogeneous scientific data using the EAV/CR representation. J Am Med Inform Assoc **6** (1999) 478–493
10. Lenz, R., Kuhn, K.A.: Aspekte einer prozessorientierten Systemarchitektur für Informationssysteme im Gesundheitswesen. In Dadam, P., Reichert, M., eds.: Informatik 2004 – Informatik verbindet, Band 2, Beiträge der 34. Jahrestagung der Gesellschaft für Informatik e. V. (GI). (2004) 530–536

Rekonstruktion einer Patientenakte und der Aufbau eines institutionsübergreifenden Wissenssystems

Brigitte Eller, Erich Ortner

Institut für Wirtschaftsinformatik
Hochschule Liechtenstein
Fürst-Franz-Josef-Strasse
FL-9490 Vaduz, Liechtenstein
E-Mail: brigitte.eller@hochschule.li

Fachgebiet Wirtschaftsinformatik I
Institut für Betriebswirtschaftslehre
Technische Universität Darmstadt
Hochschulstraße 1
D-64289 Darmstadt
E-Mail: ortner@winf.tu-darmstadt.de

Abstract. Die Erstellung eines Dokumentationsrahmens, in seinen Ausprägungen laufend anpassbar an das jeweilige Einsatzgebiet (z.B. medizinisches Fachgebiet), erscheint im Gegensatz zu einem „Ein-Welt-Modell“ richtungweisend für die Realisierung einer elektronischen Patientenakte als Basiskonstrukt eines institutionsübergreifenden Wissenssystems im Gesundheitswesen. In der sprachkritischen Systementwicklung wird semantische Integration bereits in einer frühen Phase des Entwicklungsprozesses instrumentalisiert – wahrscheinlich eine der wesentlichsten Voraussetzungen, um die Gebrauchsfähigkeit von Wissen (Knowledge-Usability) nachhaltig zu verbessern.

1 Einleitung

Der geschätzte Effizienzverlust von 15 - 20 %, welcher durch diverse Schnittstellenproblematiken (Zustand der Desintegration) innerhalb des betrachteten Gesundheitswesens (als Modellregion diente Vorarlberg in Österreich) gegeben ist [2], sollte mit einer elektronischen Patientenakte innerhalb einer integrativen Wissenssystemarchitektur erheblich reduziert werden können. Im Sinne von Transdisziplinarität - und nicht resultierend aus ev. vorhandenen Defiziten alternativer Ansätze - wird die erweiterte Methode der sprachkritischen Systementwicklung zur exemplarischen Rekonstruktion einer Patientenakte eingesetzt. Als Integrationsmedium zur Zusammenführung von Wissen aus heterogenen Systemen soll die natürliche Sprache verwendet werden, da der Sprache Integration immanent ist. Demnach ist Integration nicht nur als Architekturaspekt sondern auch als (System-)Prozess aufzufassen, da Sprache einem ständigen Evolutionsprozess unterliegt [11].

2 Grundlagen

2.1 Architektur einer Sprache

In Wissenssystemen kommen in der Regel mehrere Architekturen zum Tragen. Angelehnt an die Drei-Schema Architektur für Datenbanken nach ANSI/SPARC lässt sich für eine Sprache allgemein und für die sprachkritische Systementwicklung im Besonderen eine Spracharchitektur ableiten (Fig. 1). Die Verbindungen zwischen den Ebenen haben in der Spracharchitektur den Charakter einer Konstruktion. Aussagen können gleichermaßen wie Wissen aus Sprache (re-)konstruiert werden bzw. durch Sprache begründet werden. Die gewählte Sprache stellt dabei das integrative Element dar, welches auf Strukturen (Grammatik, Satzbau) und Begriffe (Lexika, geklärte Fachbegriffe) zurückgreift.

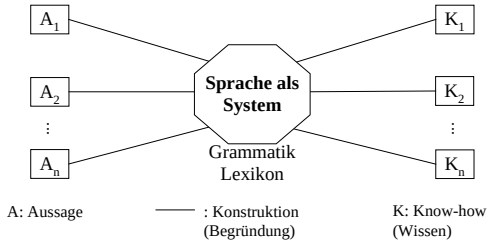

Fig. 1. Spracharchitektur

Die Integration über Sprache erfolgt mittels des materialsprachlichen Ansatzes (= Sprache als System), wie er von Ortner [8] beschrieben wurde. Durch Erarbeitung und Einbeziehung von systematischen Sprachprodukten (z.B. Bereichsontologien - vgl. Abschn. 2.4) in einer frühen Phase der Systementwicklung wird die Basis für die angestrebte, nachhaltig wirksame, semantische Integration geschaffen. Zum Zweck des besseren Managements des Entwicklungsprozesses erfolgt eine phasenorientierte Einordnung in das Multi-Pfad-Vorgehensmodell [11].

2.2 Architektur eines Wissenssystems

Wissen ist nach Wittgenstein ein besonderes Können. Es handelt sich um „geistige Anlagen“ die als Schema repräsentiert und verwendet werden (Ability-Management). Für den Menschen ist sein Wissen, für den Rechner ist die Software das Schema nach dem er arbeitet. Die Synthese von beiden bildet die rechnerunterstützte Wissensverarbeitung (Wissensarbeit). Bereits Kant nannte das Verfahren (z. B. beschrieben in Form einer Anleitung zur Konstruktion eines rechtwinkeligen Dreiecks mittels Zirkel und Lineal) einem Begriff (im Beispiel das rechtwinkelige Dreieck) sein „Bild“ (Gegenstand) zu verschaffen, das Schema eines Begriffs. Ein Schema ist demnach der universelle (abstrakte) Aspekt eines Objekts (Ding oder Geschehnis) - Schemata akkumulieren Wissen. Eine Ausprägung ist demgegenüber der singuläre Aspekt eines Objekts - Ausprägungen transportieren Information. Diese

grundsätzlichen Zusammenhänge auf Metaebene zeigt Fig.2 nach [5] in Form einer Architektur eines Wissenssystems, wie sie institutionsübergreifend im Gesundheitswesen realisiert werden kann.

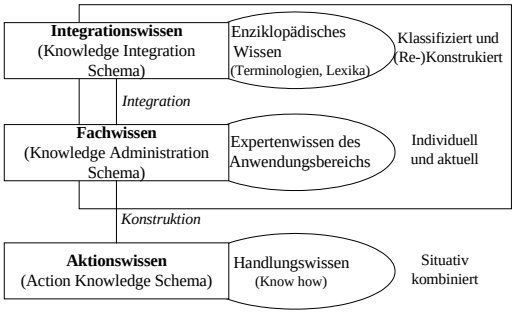

Fig. 2. Architektur eines Wissenssystems

Mit dem Aktionswissen (Action Knowledge Schema) wird problemspezifisch bzw. situativ eingesetztes Expertenwissen aus den Anwendungsbereichen beschrieben (Knowledge Administration Schema). Die Zusammenhänge innerhalb dieses Fachwissens werden repräsentiert durch das Integrationswissen (Knowledge Integration Schema). Wissensarbeit lässt sich als umfassende Integration von Wissen aus verschiedensten Disziplinen sehen, umgekehrt aber nicht in einige Expertenbereiche aufteilen.

2.3 Erweiterte Methode des normsprachlichen Entwurfs

Die Methode des normsprachlichen Entwurfs wird erweitert um das Usability-Engineering zur Rekonstruktion von Aktions- und Fachwissen im Anwendungsbereich um Knowledge-Usability zu unterstützen. Die Vorgehensschritte der auf Normalsprache (Gebrauchssprache) basierenden Modellierung lassen sich nach [11] wie folgt auflisten und werden unter dem Begriff „sprachkritische Systementwicklung“ subsumiert:

1. Sammlung relevanter Aussagen (Kommunikation über Anwendungsbereiche zwischen Entwickler und Anwender in natürlicher Sprache)
2. Klärung und Rekonstruktion der Fachbegriffe (Rekonstruktion von Sprache und Sprachartefakten des Anwendungsbereiches durch Verwendung von Lexika und Satzbauplänen auf Basis der Prosaaussagen - semantische Modellierung)
3. Ableitung von Schemata und Übergang zu einem klassifizierten Aussagenbestand (Modifikation und Transformation hin zu reglementiertem Satzbau sowie Klassifizierung)
4. Diagrammsprachliche Darstellung (Spezifikation der normierten Aussagen mit einer Kunstsprache)

Durch das schrittweise, wenn erforderlich iterative, systematische Vorgehen wird sichergestellt, dass die Ergebnisse der Entwicklungsarbeit nicht einfach „vom Himmel“ fallen (dogmatisch) oder „intuitiv“ (naiv) erzielt, sondern (re-)konstruiert werden [6].

2.4 Medizinische Ordnungssysteme und Inhaltsstandards

Der Begriff „Inhaltsstandards“ wird hier zweigeteilt verstanden. Einerseits geht es um Standards für die Darstellungsform und Anordnung von Inhalten (z.B. CDA, open(G)EHR - siehe Abschn. 4). Andererseits handelt es sich um Standards z.B. zur Festlegung und Verbindung medizinischer Fachtermini (z.B. SNOMED, ICD), d.h. auch die Einbeziehung von Nomenklaturen, Thesauri und Ergebnissen von Integrationsprojekten verschiedener medizinischer Ordnungssysteme (z.B. GALEN [13], UMLS als semantisches Netzwerk [3], ONIONS [12]) wird hier unter diesem Begriff subsumiert. Im dargestellten Ansatz spielt die Erarbeitung, Erweiterung und Integration vorhandener Inhaltsstandards in Form vorhandener Nomenklaturen, Thesauri und Ontologien ab einer sehr frühen Phase der Systementwicklung eine wesentliche Rolle, wobei eine die laufende Bewältigung sowohl der intensionalen als auch der extensionalen semantischen Integration (während des gesamten Systemlebenszyklus) methodisch unterstützt werden muss. Colomb [4] spricht in diesem Zusammenhang von struktureller und fundamentaler semantischer Integration.

3 Fachentwurf

3.1 Rekonstruktionsprozess und -methoden

In der Phase des Fachentwurfs stehen Methodenneutralität, Normsprachlichkeit und deren materialer Charakter (Materialsprachlichkeit) und damit verbunden die semantische Integration im Vordergrund. Die Rekonstruktion von Sprache und Sprachartefakten des Anwendungsbereiches (z.B. Organisationsstrukturen und Anwendungswissen) erfolgt mittels der erweiterten Methode des normsprachlichen Entwurfs. Das Ergebnis der Rekonstruktion ist grundsätzlich zweigeteilt und besteht aus einer Terminologie (rekonstruierte Fachsprache in Verbindung mit etablierten Inhaltsstandards - siehe Abschn. 2.4) und normierten Aussagen.

„Fachsprachliche Rekonstruktion“ bedeutet Erarbeitung und Beschreibung systematisch rekonstruierter Sachzusammenhänge im Anwendungsbereich in normierter Form auf Basis normalsprachlicher Aussagen der Anwender [7]. Innerhalb des Rekonstruktionsprozesses gelangt der Entwickler zu einer Sammlung relevanter Aussagen in einer normierten Sprache auf Basis geklärter Fachbegriffe. Zu rekonstruierende Fachwörter (Themenwörter aus dem Anwendungsgebiet z.B. „Behandlung“) werden in der Aussagensammlung identifiziert und unter Einsatz ausgewählter Methoden (z.B. explizite Definition, Beispiele und Gegenbeispiele, Begriffsklärung durch Besiedelung eines Begriffsschemas – entspricht intensionaler und extensionaler Erschließung des Begriffs) geklärt. Auf der Begriffsklärung liegt der Schwerpunkt im Rekonstruktionsprozess. Begriffsdefekte (z.B. Synonyme, Homonyme, Vagheiten) werden erkannt, behandelt und dokumentiert. Ergänzend zur Begriffsklärung bietet sich im medizinischen Bereich die Festlegung von zulässigen bzw. priorisierten „Wertebereichen“ innerhalb der integrierten Ontologien an. Die Ergebnisse der Begriffsklärung sowie festgelegte Vorzugsbezeichnungen fließen in

eine Terminologie, die in der zugehörigen „Lexikon-Komponente“ eines Repository [9] integriert ist und dort im Systemlebenszyklus administriert wird.

3.2 Aussagensammlung und Normierung der Aussagen

Aussagen zum Anwendungsbereich werden dialogisch und hermeneutisch in Interviews, Besprechungen, Recherchen in Fachliteratur und Organisationsunterlagen, aber auch durch Beobachtungen in der Praxis erstellt. Nach der Selektion jener Aussagen, die den erforderlichen Sachbezug haben, die hinreichend genau sind und die uneingeschränkte Zustimmung der Vertreter des Anwendungsbereiches finden liegt die Aussagensammlung vor.

Neben Termini sind Gegenstandseinteilungen und Satzbaupläne (Re-)Konstruktionsmittel für Sprache. Im Zuge der Normierung werden Aussagen zum Anwendungsbereich zu einfachen Sätzen (reglementierte Sprache) transformiert. Grundlage der Transformation sind Satzbaupläne. Diese repräsentieren den Dokumentationsbereich „Grammatik“ einer Sprache. Eine detaillierte Beschreibung von Satzbauplänen auf Basis einer Gegenstandseinteilung, wie sie hier verwendet wird, findet sich bei [7]. Neben Termini und normierten Aussagen ergeben sich während des Rekonstruktionsprozesses Erkenntnisse (z.B. organisatorische Normen, Integritätsbedingungen), die nicht unmittelbar im Systementwurf verwendet werden. Diese werden dokumentiert und ebenfalls in ein Repository integriert. Auf Basis der erarbeiteten Terminologie und mit den verwendeten Nomenklaturen (diese bilden zusammen die „Lexikon-Komponente“) repräsentieren die normierten Aussagen methodenneutral den Fachentwurf.

4 Strukturierungselemente für Systemarchitekturen und Repository

Zusätzlich zu Strukturierungselementen wie Programming in the large und Programming in the small sollten für die Gestaltung einer elektronischen Patientenakte Sprachschichten und orthogonal dazu Sprachräume eingesetzt werden. Damit können bereits in der Phase des Fachentwurfs grobe Strukturen für „Benutzt-Relationen“ zwischen den Sprachschichten als Rahmen für Benutzer- und Rollenkonzepte geschaffen werden. Fig. 3 zeigt eine mögliche Organisation von Sprachschichten und Sprachräumen in einer elektronischen Patientenakte. Die Sprachräume sind ausschließlich thematisch orientiert und ermöglichen eine noch stärkere anwendungsorientierte Differenzierung, als dies bei den Sprachschichten möglich ist. Für die Anwendungsbereiche einer elektronischen Patientenakte bietet sich dadurch bereits im Fachentwurf, spätestens jedoch im Systementwurf, die Möglichkeit z.B. Datenschutzfragen hinsichtlich der besonders sensiblen medizinischen Patientendaten konzeptionell zu berücksichtigen.

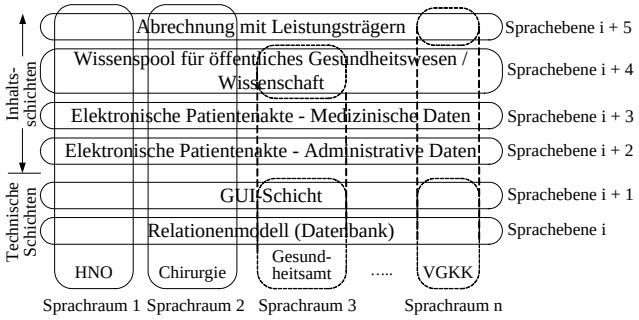

Fig. 3. Sprachebenen und Sprachräume zur Bestimmung einer Architektur der elektronischen Patientenakte

Neben anderen Vorteilen begünstigt die vorgeschlagene Strukturierung eine schrittweise Realisierung einzelner Systemfunktionalitäten. Vergleichsweise könnten Archetypen nach dem open(G)EHR Ansatz [1] als Inhaltsstandards im Sinne von Darstellungsform und Anordnung der Inhalte in den Sprachschichten bzw. Sprachräumen verstanden werden.

Jene Komponenten eines Wissenssystems (Grammatik, Lexikon, Fachentwurf, Architektur eines Wissenssystems) die erforderlich sind, um dieses sprachlich komplett nachzubilden, sind in Fig. 4 zum Dokumentationsrahmen eines Repository gem. [5], [10] für die Verwaltung von Wissen zusammen gefasst.

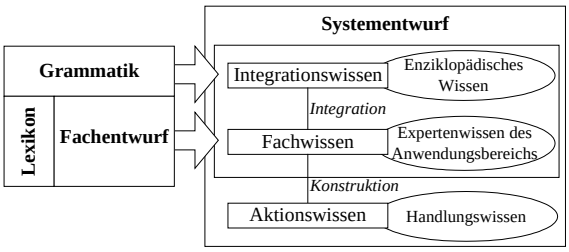

Fig. 4. Dokumentationsrahmen eines Repository zur Verwaltung von Wissen

5 Aspekte zum Systementwurf einer Patientenakte

Der Übergang vom Fachentwurf zum Systementwurf erfolgt durch Klassifizierung des Aussagenbestandes. Sind Grammatik und Terminologie der Aussagen klar definiert, klassifiziert und digital verfügbar, ist es mit einem regelbasierten System möglich, die Überführung der normierten Aussagen in eine Diagrammsprache zu automatisieren.

Die Überwindung der verschiedenen Heterogenitäten (technisch, semantisch und pragmatisch) ist innerhalb des Systementwurfs sicherzustellen. Die Voraussetzungen zur Überwindung der semantischen Heterogenität werden im Rahmen der sprachkritischen Systementwicklung bereits in den Phasen Voruntersuchung und

Fachentwurf geschaffen. Die technische Integration der erarbeiteten Terminologien kann z.B. durch Realisierung der Sprachschicht „Terminologien“ als eine oder mehrere Komponenten in der Middleware des Systems erfolgen. Die Überwindung der pragmatischen Heterogenität (= unterschiedliches Verständnis über das Handeln der jeweils an einer Transaktion beteiligten Versorgungseinrichtungen) ist Voraussetzung für eine reibungslose Abwicklung der für den Betrieb einer elektronischen Patientenakte notwendigen Transaktionen (Datensendung und Datenempfang). Lösungsansätze hierzu sind einerseits die Definition eines Interaktionsprotokolls für jede mögliche Interaktion, andererseits eine automatische Terminologiekontrolle innerhalb ausgewählter Kommunikationsvorgänge auf Basis der erarbeiteten Terminologien.

6 Fazit und Ausblick

Die Erstellung eines „Ein-Welt-Modells“ für alle Anspruchsgruppen im Anwendungsbereich würde Jahre dauern und im Vorfeld erhebliche Kosten verursachen. Ein realistischer Ansatz für ein regional begrenztes, patientenzentriertes, institutionsübergreifendes Wissenssystem liegt in der Zusammenführung der Informationen auf eine gemeinsamen Plattform unter Beibehaltung der heterogenen Systemlandschaft in den Anwendungsbereichen. Die permanente Bewältigung (nicht die Beseitigung) der semantischen Heterogenität (beispielsweise durch automatische Übersetzung des Informations-Input auf Basis eines Repository Systems [9], [10] mit dargestelltem Dokumentationsrahmen) ermöglicht die Schaffung dieser integrierten Datenbasis. Die Ergebnisse der exemplarisch durchgeführten Rekonstruktion einer Patientenakte nach beschriebener Methode machen deutlich, dass der Ansatz richtungweisend zur Realisierung einer nachhaltig integrativen Lösung ist, wobei auch den Forderungen nach Flexibilität und Rationalität des Wissenssystems Rechnung getragen werden kann. Inwieweit die Gebrauchsfähigkeit des Wissens (Knowledge-Usability) für verschiedenste Nutzer einer solchen Plattform sichergestellt werden kann, bleibt noch zu untersuchen.

Besonders hervorhebenswert ist die Möglichkeit, mittels einer rekonstruierten in einem Repository hinterlegten „objekt- und metasprachlichen Normsprache“ einen Strategiewechsel von einer referenzmodellorientierten zu einer terminologiebasierten Entwicklung und Nutzung von Informationssystemen vorzunehmen. Gerade im betrachteten Anwendungsgebiet der Gesundheitsversorgung mit ihren immer neuen Herausforderungen können Heterogenitätsprobleme durch diese sprachbasierte, semantische Integration mittels Terminologien und Ontologien statt durch integrierte Mammut-Lösungen (Ein-Welt-Modell) besser gelöst werden. Dabei wird weniger Wert auf eine „gewaltsame“ Vereinheitlichung sondern mehr Wert auf die Interoperabilität verschiedenartiger Systeme auf Grundlage rekonstruierter objekt- und metasprachlicher Terminologien („Ontologien“) gelegt.

Literaturverzeichnis

[1] Beale, T.: Archetypes and the EHR. Stud. Health Technol. Inform. 2003; 96: 238-244

[2] Bischof, H.: Neue Wege gehen. In: Österreichische Monatshefte, (4) 4-8, 2002

[3] Bodenreider, O., Nelson, S., Hole, W., Chang F.: Beyond Synonymy: Exploiting the UMLS Semantics in Mapping Vocabularies, 1998 In: http://www.amia.org/pubs/symposia/D004771.PDF, Abruf 12.05.2005

[4] Colomb R.M.: Impact of semantic heterogeneity on federating databases. In: The Computer Journal, 40 (5): 235-244, 1997

[5] Heinemann, E.; Ortner, E.; Wedekind, H.: A Framework for Language-Based Schema Management and Epistemic Application Systems. In: Proceedings of i-KNOW '04: Graz, Austria. July 2004

[6] Jablonski, S.; Böhm, M.; Schulze, W. (Hrsg.): Workflow-Management - Entwicklung von Anwendungen und Systemen. Korrigierter Nachdruck. Heidelberg: dpunkt-Verlag, 1999

[7] Ortner, Erich: Aspekte einer Konstruktionssprache für den Datenbankenwurf. Darmstadt: Toeche-Mittler, 1983

[8] Ortner, Erich: Software-Engineering als Sprachkritik. Die sprachkritische Methode des fachlichen Software-Entwurfs. Konstanzer Universitätsreden, 187. Konstanz: Univ.-Verl., 1993

[9] Ortner, E.: Repository Systems. Teil 1: Mehrstufigkeit und Entwicklungsumgebung. In: Informatik-Spektrum, 22 (4): 235-251, 1999

[10] Ortner, E.: Repository Systems. Teil 2: Aufbau und Betrieb eines Entwicklungsrepositoriums. In: Informatik-Spektrum, 22 (5): 351-363, 1999

[11] Ortner, E.: Sprachbasierte Informatik. Wie man mit Worten die Cyber-Welt bewegt. Leipzig: Edition am Gutenbergplatz, 2005

[12] Pflüglmayer, Martin: Informations- und Kommunikationstechnologien zur Qualitätsverbesserung im Krankenhaus – Computerbasierte Terminologien als semantische Basis für medizinische Informationssysteme. In: http://www.serv-it.at/Serv-IT/DownloadsFrameset.htm?Downloads.htm [Zugriff 25.07.2003]

[13] Rector A.L: Nowlan W.A..: The GALEN project. Comput. Methods Programs Biomed., 45 (1-2): 75-78, 1994

Prozessorientierte Systemintegration: Anspruch und Realität

Stephan Aier und Marten Schönherr

Technische Universität Berlin, Sekr. FR 6-7, Franklinstr. 28/29, 10587 Berlin,
stephan.aier@tu-berlin.de, schoenherr@sysedv.tu-berlin.de,
http://www.sysedv.tu-berlin.de/eai

Abstract. Großunternehmen zeichnen sich heute durch komplexe und heterogene IT-Infrastrukturen aus. Hauptziel der verantwortlichen IT-Architekten ist die zielführende Integration und Anpassung der IT an die fachlichen Anforderungen zu minimalen Kosten. Ein aktueller Ansatz zur Lösung dieser Herausforderung ist Enterprise Application Integration (EAI). In einer Studie in großen Unternehmen haben wir die Bedeutung und den Einsatz von EAI als Integrationselement von Unternehmensarchitekturen untersucht. Der Beitrag stellt Ergebnisse der Studie dar und zeigt bestehende Verbesserungspotenziale auf.

1 EAI als Antwort auf komplexe IT-Landschaften

Informationstechnologie ist zu einem bestimmenden Faktor in Unternehmen geworden. Insbesondere große Unternehmen haben in den letzten 40 Jahren komplexe und heterogene IT-Infrastrukturen aufgebaut. Integration dieser Infrastrukturen ist ein anhaltend wichtiges Thema. Das Management der dafür notwendigen Schnittstellen zwischen den IT-Systemen ist mit klassischen Punkt-zu-Punkt-Verbindungen aufgrund der Komplexität kaum noch möglich. Zum einen erzeugt der laufende Betrieb und die Wartung der oft mehreren tausend Schnittstellen hohe Kosten, zum anderen sind Anpassungen, z.B. bei Veränderung der fachlichen Anforderungen, kaum mehr möglich, da die Komplexität der Abhängigkeiten der Elemente nicht mehr beherrschbar ist [1]. Enterprise Application Integration (EAI) stellt einen Ansatz dar, der über eine meist zentrale Integrationsplattform bestehende Punkt-zu-Punkt-Verbindungen ersetzt und damit die Anzahl der Schnittstellen stark verringert. Im Gegensatz zu klassischen Middleware-Konzepten, soll EAI jedoch auch fachliche Integration auf Geschäftsprozessebene ermöglichen (Fig. 1) [2-5].

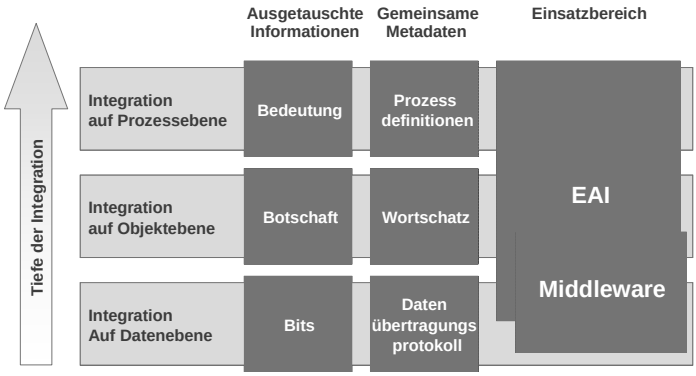

Fig. 1. Integrationsebenen nach Ring [6]

Eines der Hauptziele des EAI-Einsatzes ist es, mehr Flexibilität bei der Veränderung von Geschäftsprozessen durch eine bessere Verknüpfung mit der IT-Infrastruktur zu minimalen Kosten zu erreichen.

Im Folgenden wird die Rolle von EAI als technische und fachliche Integrationskomponente der Unternehmensarchitektur dargestellt. Anschließend werden Vorgehensweise und Ergebnisse unserer Studie erläutert, die den Einsatz und die Wirkungen von EAI in Unternehmensarchitekturen untersucht hat. Darauf aufbauend werden Potenziale für eine bessere Integration von Unternehmensarchitekturen aufgezeigt.

2 EAI als Kernelement der Unternehmensarchitektur

Allgemein kann eine Architektur als eine abstrakte, ganzheitliche Betrachtung von Strukturen und Mustern mit Planungscharakter aufgefasst werden [7, 8]. Speziell wird unter einer Unternehmensarchitektur das Zusammenwirken organisatorischer, technischer und psychosozialer Aspekte bei der Planung und Entwicklung betrieblicher sozio-technischer Informationssysteme verstanden [9]. Im Folgenden sollen vor allem die organisatorische und die technische Dimension der Unternehmensarchitektur betrachtet werden. Dazu werden die Begriffe *Organisationsarchitektur* und *IT-Architektur* verwendet (Fig. 2).

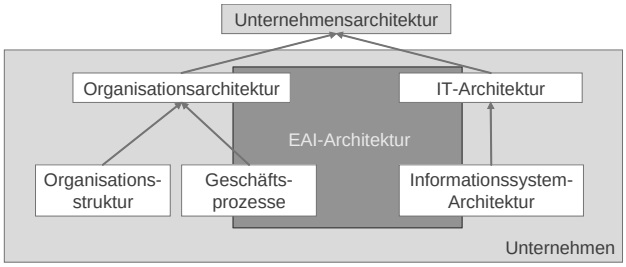

Fig. 2. Bestandteile der Unternehmensarchitektur [10]

Die *Organisationsarchitektur* enthält alle nichttechnischen Bestandteile der Unternehmensarchitektur und kann mit dem instrumentalen Organisationsbegriff verglichen werden, der die Gesamtheit aller generellen expliziten Regelungen zur Gestaltung der Aufbau- und Ablauforganisation umfasst [11]. Entsprechend wird die Organisationsarchitektur hier in die Organisationsstruktur und die Geschäftsprozesse unterschieden.

Komplementär zur Organisationsarchitektur steht die *IT-Architektur*. Sie umfasst alle technischen Bestandteile der Unternehmensarchitektur. Insbesondere beinhaltet sie die technischen Informationssysteme, welche ihrerseits eine eigene Architektur, die Informationssystemarchitektur, aufweisen.

Im Folgenden sollen beide Teilarchitekturen separat und somit bezogen auf ihre Bedeutung gleichwertig betrachtet werden. Dies ist sinnvoll, da beide eine hohe Bedeutung für das effiziente Funktionieren einer Unternehmung haben und zwischen ihnen komplexen Wechselwirkungen bestehen [2].[1]

Per Definition hat EAI insofern eine Vermittlerstellung in der Unternehmensarchitektur, als dass EAI zum einen die technischen Elemente (IT-Systeme) als auch technische mit fachlichen Elementen (Geschäftsprozesse) verbinden soll.

3 Ziele und Vorgehensweise der Studie

EAI-Projekte verlaufen in der Praxis sehr unterschiedlich. Themen wie Toolauswahl, Einführungsmethoden, Geschäftsprozessorientierung, Verantwortung und strategische Positionierung im Unternehmen und Betrieb im Zusammenhang mit EAI-Konzepten werden je nach Kontext sehr unterschiedlich verstanden und umgesetzt. Ziel der Studie war es, zu untersuchen, wie große Unternehmen EAI tatsächlich einsetzen und welche Auswirkungen der EAI-Einsatz auf Unternehmensarchitekturen hat.[2] Es stellt sich also die Frage, ob EAI dem theoretischen Anspruch – und dem Anspruch von Toolherstellern und Beratern – der prozessorientierten Systemintegration in der Praxis gerecht wird.

Die dazu durchgeführte empirische Studie hat demnach einen deskriptiv-explorativen Charakter, d.h. das Ziel war zum einen die Beschreibung der Ist-Situation sowie die Identifikation von darin befindlichen Strukturen, die weiter zu Hypothesen verdichtet wurden [20, 21].[3] Die Untersuchung stellt eine nicht-experimentelle Querschnittserhebung zu einem konkreten Zeitpunkt dar. Als Erhebungsmethode wurde die schriftliche Befragung per Fragebogen gewählt. Aufbauend auf den hieraus erhal-

[1] In der Literatur sind die Begriffe Organisationsarchitektur und IT-Architektur häufig verwendet, jedoch meist anders definiert. Abhängig von der fachlichen Herkunft des Autors umfasst die Organisationsarchitektur auch technische Komponenten [12] bzw. die IT-Architektur auch organisatorische Fragestellungen [13].

[2] Ähnliche Untersuchungen über die Wirkungen zwischen IT und Organisation gab es in der Vergangenheit viele. Dabei wurde oft auch eine Abhängigkeit von der konkreten IT festgestellt. In dieser Tradition versteht sich auch diese Studie [14-19].

[3] Aufgrund der bislang fehlenden theoretischen und empirischen Fundierung des Themas haben wir es für notwendig gehalten, zuerst eine deskriptiv-explorative Basisuntersuchung durchzuführen [22]. Erst in einem nächsten Schritt ist eine explanative (konfirmative) Untersuchung zur Bestätigung der Hypothesen denkbar, da es bis jetzt kein ausreichend solides Fundament für eine solche Untersuchung gab [23].

tenen Ergebnissen einer überwiegend deskriptiv-quantitativen Analyse wurde eine zweite Befragungsrunde durchgeführt, die entweder in persönlichen Interviews oder in Workshops mit mehreren EAI-Experten stattfand. Ziele dieser zweiten Befragungsrunde waren die Einschätzung und die Diskussion der quantitativen Ergebnisse hinsichtlich möglicher zugrunde liegender Zusammenhänge.

Die Studie wurde in mehreren Untersuchungsgruppen durchgeführt. In einem ersten Schritt wurden zunächst Unternehmen befragt, die bereits EAI-Systeme einsetzen oder in ihren Planungen zur Implementierung eines solchen Systems sehr weit fortgeschritten sind (Anwender). Während der laufenden Erhebung unter den Anwendern wurde die Umfrage um zwei weitere Gruppen von Befragten erweitert. Es wurden einerseits IT-Beratungen hinzugezogen, die Leistungen um das Thema EAI anbieten und aktiv an Projekten bei Anwendern beteiligt sind. Andererseits wurden auch Softwareunternehmen befragt, die Tools im Kontext von EAI erstellen (Toolanbieter).

Insgesamt wurden 104 Anwender, 109 Berater und 84 Toolanbieter für die Studie ausgewählt und per Fragebogen kontaktiert. Dieses Vorgehen führte zu Rücklaufquoten von rund 30,8% bei den Anwendern, 8,3% bei den Beratern und 21,4% bei den Toolanbietern.

Die zweite Befragungsrunde fand im Rahmen der 3. EAI-Expertentage an der TU Berlin statt. Es wurde ein Workshop mit 17 Teilnehmern durchgeführt, auf dem zunächst die Ergebnisse der deskriptiven Analyse vorgestellt und anschließend diskutiert wurden. Weiterhin fanden auch die Einzelinterviews statt. Die Diskussionen wurden jeweils in detaillierter Form protokolliert.

Die hieraus gewonnenen Erkenntnisse wurden anschließend zum einen einer explorativ-qualitativen Inhaltsanalyse und zum anderen einer explorativen, also strukturentdeckenden Faktor- und Hauptkomponentenanalyse [24] unterzogen. Beide parallel und unabhängig voneinander vollzogenen Analyseschritte ergaben korrespondierende Ergebnisse, die zu den im Folgenden vorgestellten Hypothesen verdichtet wurden.

4 Ergebnisse der Studie

Verständnis und Rolle von EAI: EAI wird als eine etablierte Technologie verstanden, die in großen Unternehmen Punkt-zu-Punkt-Schnittstellen und klassische Middleware ablöst. Sie ist eine langfristige, strategische IT-Architekturkomponente. Weiter ist EAI ein modularer Werkzeugkasten der im Wesentlichen Adapter, Transformationswerkzeuge, Monitoring- und Workflow-Werkzeuge und Business Process Management umfasst. Die Bedeutung von EAI als Architekturkomponente wird aus zwei Zahlen deutlich: Etwa zwei Drittel aller IT-Systeme der Befragten Unternehmen sollen über EAI verbunden werden, über die Hälfte der zu verbindenden Systeme sind bereits angebunden (Fig. 3).

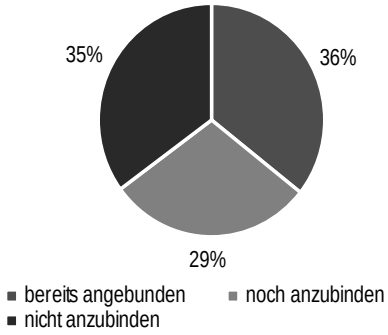

Fig. 3. An EAI anzubindende und bereits angebundene IT-Systeme

Organisation der EAI: EAI-Einführungsprojekte sind beratungsintensiv, wohingegen EAI als IT-Architekturelement von wenigen IT-Mitarbeitern betrieben wird. Der Blick auf EAI ist IT-zentrisch, nicht ganzheitlich. Erweiternd kann gesagt werden, dass der Hauptauslöser für die Beschäftigung mit EAI die Integration neuer IT-Systeme und der damit verbundene Aufwand sind.

Prozessorientierung und EAI: Das Ziel, EAI prozessorientiert einzusetzen, wird in der Praxis kaum erreicht. Wenn von einer Integration auf Prozessebene gesprochen wird, meint man oft technische Prozesse, keine Geschäftsprozesse (Fig. 4). Dennoch löst EAI Prozessänderungen aus. Im Detail beutet das: EAI soll bessere, effizientere und schnellere Prozesse schaffen. Die wichtigsten Prozessänderungen mit der EAI-Einfürung sind Automatisierung und Konsolidierung. Diese Prozessänderungen führen zu einer höheren Formalisierung der Organisationsarchitektur. Weiter hat EAI eine indirekte Enabler-Wirkung auf fachliche Prozesse, d.h. mit der EAI-Einführung werden bestimmte fachliche Aufgaben erst möglich. Trotz dieser deutlichen Wirkungen auf die Geschäftsprozesse ist Prozessmanagementfunktionalität kein wichtiges Kriterium bei der EAI-Produktauswahl auch wird EAI kaum zur Erhebung fachlicher Leistungsdaten benutzt.

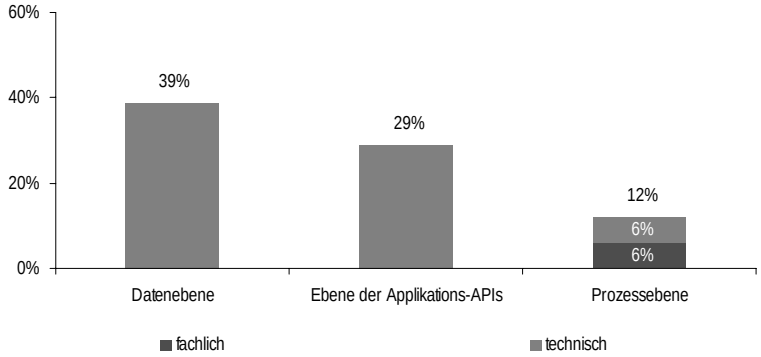

Fig. 4. Auf welcher Ebene integriert die Plattform die Anwendungen? (Nur EAI-Anwender)

Aufbauorganisation und EAI: EAI führt kaum zu Änderungen der Aufbauorganisation. Stattfindende Änderungen vor allem eine stärkere Formalisierung und Dokumentation der Organisationsarchitektur. EAI-Abteilungen etablieren vor allem

mentation der Organisationsarchitektur. EAI-Abteilungen etablieren vor allem technische, weniger organisatorische Richtlinien. Die EAI-Verantwortlichen machen nur undifferenzierte Aussagen zur Organisationsarchitektur.

5 Fazit

Zusammenfassend muss gesagt werden, dass EAI dem Anspruch der zentralen technischen Integrationsplattform heute und in Zukunft gerecht wird. Jedoch scheitert EAI bei dem eigentlich neuen Anspruch der Integration auf Prozessebene und damit der Integration von IT und Organisation in einer Unternehmensarchitektur. Unseres Erachtens nach liegt das nur zum Teil am Ansatz von EAI selbst. Wichtiger ist, wie EAI derzeit in Unternehmen verstanden wird. Die Mehrheit betreibt EAI als technische Integrationsinfrastruktur in einer primär technisch denkenden IT-Abteilung. Notwendig jedoch wäre der Betrieb durch eine zentrale, übergreifende Architektureinheit die sich als strukturierender Moderator zwischen IT und Fachbereichen versteht. Diese auch als Architekturmanagement formulierte Funktion stellt damit eine logische Schlussfolgerung als Antwort auf die Ergebnisse der Studie dar. Einem ganzheitlichen Architekturmanagement liegt ein entsprechendes Architekturverständnis zugrunde, das eine durchgängige Betrachtung von fachlichen Anforderungen und Abläufen hin zur IT-Infrastruktur und strategischen Technologieentscheidungen voraussetzt. Wichtiger Bestandteil der beschriebenen Funktion ist ein individuelles Set von architekturbestimmenden Methoden, Modellierungsnotationen und Tools, die über die durch die Studie aufgezeigten Barrieren fachbestimmten Denkens hinausgehen und die relevanten Sichten verbinden. Hier stehen u.a. Lebenszyklusaspekte der modellierten Architektur im Vordergrund. Weitere Herausforderung ist die stringente Definition von Zielen und deren Verfolgung durch permanent etablierte Verantwortlichkeiten, die nicht projektbezogen evaluieren sondern die langfristigen Eigenschaften der Gesamtunternehmensarchitektur betrachten. Dafür gibt es wie beispielsweise [1] zeigt erfolgreiche Beispiele.

Literatur

1. Hagen, C. Integrationsarchitektur der Credit Suisse. In: Aier, S., Schönherr, M. (Hrsg.) Enterprise Application Integration – Management komplexer Architekturen. Gito, Berlin (2003) S. 61–83
2. Aier, S., Schönherr, M. Flexibilisierung von Organisations- und IT-Architekturen durch EAI. In: Aier, S., Schönherr, M. (Hrsg.) Enterprise Application Integration – Management komplexer Architekturen. Gito, Berlin (2003) S. 3–56
3. Kaib, M.: Enterprise Application Integration: Grundlagen, Integrationsprodukte, Anwendungsbeispiele. DUV, Wiesbaden (2002)
4. Keller, W.: Enterprise Application Integration – Erfahrungen aus der Praxis. Dpunkt, Heidelberg (2002)
5. Linthicum, D.S.: Enterprise Application Integration. Addison-Wesley Longman, Amsterdam (2000)
6. Ring, K.: EAI: Making the right Connections. Boston (2000)

7. Bass, L., Clements, P., Kazman, R.: Software Architecture in Practice. 2. Auflage, Pearson Education Inc., Boston (2003)
8. Wall, F.: Organisation und betriebliche Informationssysteme – Elemente einer Konstruktionslehre. Gabler, Wiesbaden (1996)
9. Gronau, N.: Wandlungsfähige Informationssystemarchitekturen – Nachhaltigkeit bei organisatorischem Wandel. Gito, Berlin (2003)
10. Aier, S., Dogan, T. Indikatoren zur Bewertung der Nachhaltigkeit von Unternehmensarchitekturen. In: Ferstl, O.K., Sinz, E.J. (Hrsg.) Wirtschaftsinformatik 2005: eEconomy, eGovernment, eSociety. Physica, Heidelberg (2005) S. 607–626
11. Bleicher, K.: Organisation: Strategien, Strukturen, Kulturen. 2. Auflage, Gabler, Wiesbaden (1991)
12. Nadler, D.A., Gerstein, M.S., Shaw, R.B.: Organisationsarchitektur. Campus, Frankfurt/Main, New York (1994)
13. Krcmar, H.: Bedeutung und Ziele von Informationssystem-Architekturen. In: Wirtschaftsinformatik 32 (1990) S. 395–402
14. Markus, M.L., Robey, D.: Information Technology and Organizational Change: Causal Structure in Theory and Research. In: Management Science 34 (1988) S. 583–589
15. Lewin, A.Y., Hunter, S.D. Information Technology & Organizational Design: A Longitudinal Study of Information Technology Implementations in the U.S. Retailing Industrie, 1980–1996. In: Glaser, H., Schröder, E.F., Werder, A.v. (Hrsg.) Organisation im Wandel der Märkte. Gabler, Wiesbaden (1998) S. 251–286
16. Rockart, J.F., Short, J.E.: IT in the 90's: Managing Organizational Independence. In: Sloan Management Review (1989) S. 7–17
17. Burgfeld, B.: Organisationstheorie und Informationstechnologie. DUV, Wiesbaden (1998)
18. Teubner, R.A.: Organisations- und Informationssystemgestaltung. DUV, Wiesbaden (1999)
19. Leavitt, H.J., Whisler, T.L.: Management in the 1980s: New Information Flows Cut New Organization Flows. In: Harvard Business Review 36 (1958) S. 41–48
20. Diekmann, A.: Empirische Sozialforschung – Grundlagen, Methoden, Anwendungen. 12. Auflage, Rowohlt Taschenbuch Verlag, Reinbek (2004)
21. Bortz, J., Döring, N.: Forschungsmethoden und Evaluation: für Human- und Sozialwissenschaftler. 3. Auflage, Springer, Berlin, Heidelberg et al. (2002)
22. Kromrey, H.: Empirische Sozialforschung. 10. Auflage, Leske + Budrich, Opladen (2002)
23. Tukey, J.W.: Exploratory Data Analysis. Addison-Wesley Publishing Company, Reading (Massachusetts) (1977)
24. Backhaus, K., Erichson, B., Plinke, W., et al.: Multivariate Analysemethoden. 10. Auflage, Springer, Berlin, Heidelberg et al. (2003)

Integration von prozessorientierten Anwendungen

Thomas Bauer

DaimlerChrysler Research and Technology
Abt. Data and Process Management
Postfach 2360, D-89013 Ulm
Thomas.TB.Bauer@DaimlerChrysler.com

Zusammenfassung. Existierende Altanwendungen enthalten oft explizites oder implizites Prozesswissen. Ziel muss es sein, dieses bei ihrer Integration in eine unternehmensweite Anwendung verfügbar zu machen und zu nutzen. Im vorliegenden Beitrag wird untersucht, welche Ansatzpunkte und Anforderungen es hierfür gibt. Von diesen ausgehend wird eine entsprechende systemneutrale Architektur und eine exemplarische Lösung für IBM WebSphere Business Integration (WBI) vorgestellt.

1 Einleitung

Ein wichtiger Aspekt der Enterprise Application Integration (EAI) sind Prozesse. Diese werden meist verwendet, um existierende Anwendungen zu integrieren (siehe z.B. [2]). Eine in der wissenschaftlichen Literatur bisher kaum betrachtete Tatsache ist, dass die zu integrierenden Altanwendungen häufig selbst Prozesslogik enthalten. Ziel muss es sein, dieses Prozesswissen explizit zu machen, um es in neu zu erstellenden (übergeordneten) Applikationen nutzbar zu machen. Das bedeutet, die Altanwendung soll nicht mehr nur als „Black Box“ eingebunden werden.

Dies ist insbesondere deshalb eine Herausforderung, weil existierende Legacy Applications normalerweise nicht wohl-strukturiert z.B. auf Basis von Workflow-Management-Systemen (WfMS) [7] realisiert sind. Außerdem verbieten sich deren Neuimplementierung oder umfangreiche Anpassungen häufig aufgrund ihrer komplexen Geschäftslogik und aus Kostengründen. So ist der einzig verbleibende Integrationsweg, die Altanwendungen mittels eines Adapters an die neuen Applikationen anzubinden. Dieser Adapter muss natürlich Zugriff auf die benötigten Daten und Ereignisse aus der Legacy Application haben [8]. Bei Eigenentwicklungen sind zur Adapteranbindung auch geringfügige Erweiterungen der Altanwendung vorstellbar (z.B. die Protokollierung relevanter Ereignisse in einer Datenbanktabelle, auf die dann vom Adapter zugegriffen wird).

Im Abschnitt 2 werden Anforderungen an die Integration von prozessorientierten Altapplikationen vorgestellt. In Abschnitt 3 wird eine generische Integrationsarchitektur vorgestellt und exemplarisch erläutert, wie diese für eine bestimmte Systemlandschaft (WBI) realisiert werden kann. Nach einer Diskussion verwandter Ansätze in Abschnitt 4 schließt dieser Beitrag mit Zusammenfassung und Ausblick.

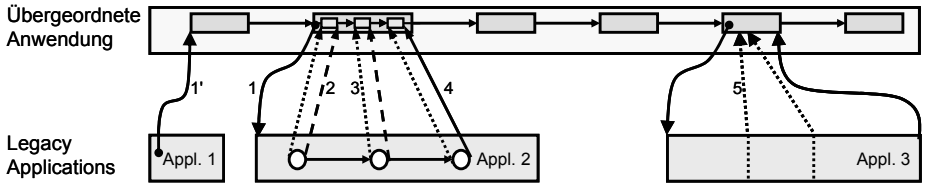

Abb. 1. Interaktionsformen zwischen Altanwendungen und der integrierenden Applikation

2 Anforderungen an die Integration von Prozessanwendungen

Im Rahmen eines Forschungsprojektes haben wir gemeinsam mit DaimlerChrysler-Unternehmensbereichen Anforderungen an eine Integration von prozessorientierten Altanwendungen gesammelt. Besonders relevant sind die funktionalen Anforderungen, da aus ihnen die notwendigen Interaktionsformen zwischen den Applikationen abgeleitet werden können. Wie in Abb. 1 dargestellt, müssen folgende Aktionen unterstützt werden:

1. Starten einer Prozessinstanz der Legacy Application durch die übergeordnete Anwendung. Hierbei müssen außer den Eingabeparametern des Prozesses auch noch der Prozesstyp und weitere Metainformationen übergeben werden. In Sonderfällen kann ein Prozess der übergeordneten Anwendung auch durch eine Aktion einer Legacy Application gestartet werden (Aktion 1' in Abb. 1).
2. Die Beendigung einer Aktivität einer Legacy Application wird an die zugehörige übergeordnete Prozessinstanz signalisiert. Diese enthält sog. „Schattenaktivitäten“, die nicht aktiv ausgeführt werden, sondern nur den aktuellen Zustand und die Ergebnisdaten der korrespondierenden externen Aktivitäten widerspiegeln. Eine solche Schattenaktivität kann in der übergeordneten Applikation z.B. als Ausgangsknoten von Kontroll- und Datenflusskanten verwendet werden. Aktivitäten können dadurch die Daten der Legacy Application nutzen, bevor diese komplett beendet ist. Die Daten von Schattenaktivitäten können außerdem für Monitoring, Prozessvisualisierung [3] und Ähnlichem genutzt werden.
3. Das Starten einer Aktivität der Legacy Application wird ebenfalls an die Schattenaktivität übermittelt. Diese Aktion kann entfallen, wenn sie in der entsprechenden Anwendung nicht erkannt werden kann (z.B. für Operationen wie „Dokument erstellen“). In diesem Fall wird der Start gemeinsam mit der korrespondierenden Aktivitätenbeendigung (Aktion 2) signalisiert.
4. Die Beendigung eines Applikationsprozesses ist für die übergeordnete Anwendung prinzipiell auch relevant. Allerdings fällt diese Aktion normalerweise mit der Beendigung seiner letzten Aktivität zusammen, so dass eigentlich eine Aktion vom Typ 2 ausgeführt wird.
5. Es gibt auch Applikationsprozesse, die nicht wirklich separate Aktivitäten durchlaufen. Dennoch kann Information über den Ausführungszustand des Prozesses anfallen, wie ein Reifegrad eines Bauteils oder ein Abarbeitungsgrad (in %). Solche Information ist für das Monitoring des übergeordneten Prozesses ebenfalls sehr relevant. Deshalb muss sie übertragen und in der jeweiligen Aktivität gespei-

chert werden. Hierbei ist essentiell, dass die Statusinformation nicht erst bei Beendigung der übergeordneten Aktivität gespeichert wird (weil die Information einen aktuellen Zwischenzustand beschreibt) und auch mehrfach aktualisiert werden kann.

Des Weiteren gibt es auch nicht-funktionale Anforderungen, die primär die Kosten und den Betrieb des Gesamtsystems betreffen:

- Für die Entwicklungskosten ist entscheidend, dass nicht (mehrfach) projektspezifische Lösungen realisiert werden. Stattdessen wird einmalig eine Infrastruktur zur Applikationsintegration entwickelt, mit der dann beliebige Altanwendungen integriert werden können.
- Um in einem Rechenzentrum einen stabilen Betrieb mit akzeptablem Aufwand garantieren zu können, darf die Infrastruktur nur ohnehin eingesetzte Serverkomponenten enthalten. Es ist nicht möglich, die Verfügbarkeit einer Applikation zu gewährleisten, die nach einem Absturz manuell gestartet werden muss. Deshalb sind z.B. Programme mit Callbacks, die von Legacy Applications gerufen werden, ausgeschlossen. Stattdessen sollten diese über Nachrichten kommunizieren, welche dann automatisch die Ausführung von Programmcode (z.B. message-driven Enterprise Java Beans) auslösen.
- Um Lizenzen und Know-how effizient nutzen zu können, muss sich die Infrastruktur auf Komponenten abstützen, die in dem jeweiligen Unternehmen Standard sind. Im vorliegenden Fall ist dies IBM WebSphere Business Integration (WBI). Außerdem muss natürlich eine geeignete Betriebssystem-Plattform ausgewählt werden.

3 Konzept und Architektur

Da IBM WBI [1, 5, 6] als Basis verwendet werden soll, wird MQ Workflow für die Realisierung von neuen Prozessanwendungen verwendet, und damit für die Implementierung des übergeordneten Prozesses (siehe Abb. 2). Zur sicheren Kommunikation dient in diesem Umfeld der Messaging-Dienst WebSphere MQ, mit dem XML-Nachrichten unter Transaktionsschutz ausgetauscht werden können. Er wird ergänzt durch den WBI Message Broker, mit welchem (automatisch ausgeführte) Microflows definiert werden können, deren Knoten u.a. folgende Operationen ermöglichen:

- Transformation eines Eingabe- in ein Ausgabedatenformat (graphisch definierbar)
- Zugriff auf Datenbanken mittels ESQL (extended SQL)
- Verzweigungen (exklusiv aufgrund von Bedingungen oder parallel)
- (inhaltsbasiertes) Routing von Messages
- Publish-/Subscribe-Mechanismus zur Registrierung für bestimmte Ereignisse

Die Ein- und Ausgabe eines solchen Microflows erfolgt stets mittels einer Message-Queue. Über eine solche kommuniziert der Message-Broker mit Adaptern. Deren Aufgabe ist es, Nachrichten entgegenzunehmen und die entsprechende Information über einen beliebigen Mechanismus (z.B. API oder Datenbankzugriff) an eine Legacy Application zu übergeben bzw. dort eine entsprechende Funktion aufzurufen. Außerdem erkennt der Adapter Ereignisse, die in der Legacy Application eingetreten sind,

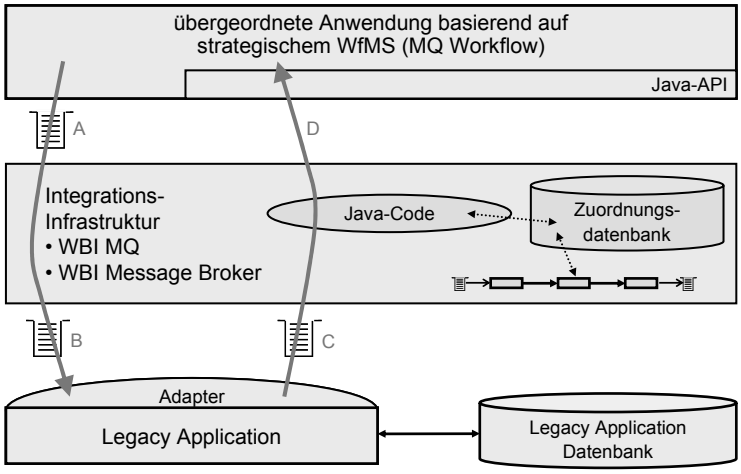

Abb. 2. Einbettung der Infrastruktur

greift auf die zugehörige Information zu, und übergibt eine entsprechende Nachricht an die Eingangs-Queue des Message Brokers.

Der eigentliche Kern der Integrations-Infrastruktur (vgl. Abb. 2) wird durch die Abläufe des Message-Brokers, Java-Code und eine Datenbank gebildet. In letzterer wird z.B. die zur Modellierungszeit spezifizierte Zuordnung einer Aktivität einer Legacy Application zu einer bestimmten Aktivität eines Workflow-Typs von MQ Workflow gespeichert. Sie verwaltet aber auch Run-Time-Daten, wie die Zuordnung von konkreten Legacy-Prozessinstanzen zu Workflow-Instanzen. Eine solche Zuordnung kann je nach Gegebenheit sowohl durch interne IDs (z.B. ID einer MQ Workflow Instanz), wie auch durch die Speicherung von Anwendungsdaten (z.B. Auftragsnummer) erfolgen.

Mit den prozessorientierten Altanwendungen und dem neuen Anwendungssystem kommuniziert die Infrastruktur über eine (intern) standardisierte Nachrichten-Schnittstelle. Dadurch wird es möglich, Adapter wiederzuverwenden und Altanwendungen schnell in unterschiedliche Prozesse zu integrieren. Die in Abb. 1 dargestellten Aktionen führen bei ihrer Ausführung in der Infrastruktur zu folgenden Operationen (mit jeweils korrespondierender Nummer):

1. Beim Starten eines Anwendungsfalls in der Altapplikation wird aus der bei der Infrastruktur eingehenden Nachricht (A in Abb. 2) ermittelt, welcher Prozesstyp in welcher Applikation gestartet werden soll. Aus der Zuordnungsdatenbank werden die entsprechende Funktion und der Adapter ermittelt, und mit den entsprechenden Eingabedaten (die zuvor ggf. vom Message Broker in das Zielformat transformiert wurden) aufgerufen (B). Das vom Adapter zurückgegebene Funktionsergebnis beinhaltet die Prozessinstanz-ID in der Legacy Application, die dann gemeinsam mit der ID aus MQ Workflow in der Zuordnungsdatenbank gespeichert wird. Analog wird verfahren, falls die Legacy Application einen übergeordneten Workflow auslöst (Aktion 1' in Abb. 1).

2. Die Beendigung einer Aktivität der Anwendung wird vom Adapter automatisch erkannt (ggf. durch Polling) und per Nachricht (C in Abb. 2) an die Infrastruktur übermittelt. Diese ermittelt aus den Modellierungsdaten der Zuordnungsdatenbank den betroffenen Workflow-Typ und die Aktivität ebenso wie die zur Ausführungszeit (durch Operation 1) hinterlegte Workflow-Instanz. Die hierdurch referenzierte MQ Workflow Aktivitäteninstanz wird dann beendet und die (evtl. transformierten) Ergebnisdaten werden übergeben (D).

   Da die Messaging-Schnittstelle von MQ Workflow die hierfür notwendige API-Funktion „CheckIn" [6] nicht unterstützt (ebenso wie die nachfolgend benötigten), muss das Java-API verwendet werden. Deshalb wird es erforderlich, in der Infrastrukturschicht Java-Code auszuführen. Wie in Abschnitt 2 beschrieben, ist es nicht möglich ist, Callbacks (z.B. per RMI) eines laufenden Java-Programms zu rufen. Deshalb werden message-driven Enterprise Java Beans verwendet, die im Falle einer eingehenden Nachricht automatisch gerufen werden und an welche die Eingabedaten übergeben werden. Diese EJB kann dann die Methode „CheckIn" des MQ Workflow Java-APIs aufrufen und die Aktivität beenden. (Eine Alternative, die ohne Application Server auskommt, ist die Verwendung einer sog. Trigger-Queue von WebSphere MQ [5].)
3. Der Start einer Aktivität der Legacy Application wird analog zur Aktion 2 erkannt und mittels der API-Funktion „CheckOut" an MQ Workflow signalisiert. Der einzige Unterschied ist, dass beim Starten einer Aktivität noch keine Ergebnisdaten entstanden sind, und deshalb nur Metainformationen (z.B. Bearbeiter, Startzeit) an MQ Workflow übermittelt werden.
4. Die Beendigung eines Legacy Application Prozesses wird bei einer Aktion vom Typ 2 mit entsprechend gesetztem Attribut an die Infrastruktur „mit signalisiert". Diese hat nun prinzipiell die Möglichkeit, den entsprechenden Zuordnungseintrag aus der Datenbank zu löschen. Hiervon wird aber abgesehen, da der Platzbedarf für eine solche Prozessinstanz-Zuordnung sehr klein ist, und der Eintrag weiterhin der Nachvollziehbarkeit und Dokumentation dient.
5. Zur Speicherung von Statusinformation einer Aktivität werden Anwendungsvariablen der übergeordneten Applikation verwendet. Diese werden hierfür reserviert und in der Zuordnungsdatenbank wird gespeichert, welche Information in welcher Variablen gespeichert wird. Signalisiert eine Legacy Application nun die Änderung eines solchen Statuswerts (C in Abb. 2), wird wieder die zugehörige Aktivitäteninstanz ermittelt (ggf. gestartet) und die Variable wird geschrieben (D).

   Problematisch ist hierbei für die konkrete Realisierung, dass MQ Workflow eigentlich nur vorsieht, Ergebnisdaten bei der Beendigung der zugehörigen Aktivität zu schreiben [6]. Da dies für einen solchen Zwischenstatus aber keinen Sinn macht, wurde im Java-Code folgendes Vorgehen implementiert: Wie in Abb. 3 dargestellt, befindet sich eine eigentlich in der Legacy Application ausgeführte MQ Workflow Aktivität im Zustand „CheckedOut". Mit der Operation „Force-Restart" ist es nun möglich, den neuen Status als Parameter in den InContainer[1] der

[1] Jede MQ Workflow Aktivität liest ihre Eingabedaten aus einem InContainer und schreibt ihr Ergebnis in einen OutContainer. Dieser kann auf die InContainer von Nachfolgeaktivitäten abgebildet werden. Über API-Funktionen [6] ist es aber jederzeit möglich, auch die Daten

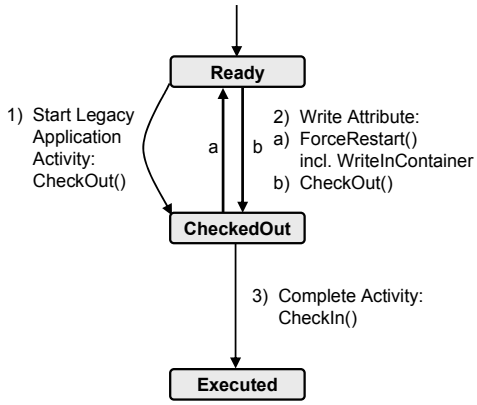

Abb. 3. Übergänge des Aktivitätenstatus bei Operation 5 (vgl. [6])

Aktivität zu schreiben. Dadurch geht der Zustand in „Ready" über. Dies macht der Java-Code durch Aufruf der API-Funktion „CheckOut" rückgängig, so dass sich die Aktivität wieder im korrekten Zustand befindet.[2] Im InContainer befindet sich dann die aktuelle Statusinformation, die durch dieses Vorgehen mehrfach aktualisiert werden kann. Beendet wird die Aktivität wie üblich mittels CheckIn.

Eine Alternative zum soeben beschriebenen Vorgehen bietet MQ Workflow durch die Definition einer Exit-Condition für die betroffene Aktivität: Beim „CheckIn" wird der Zwischenstatus in den OutContainer geschrieben und die Exit-Condition auf False gesetzt. Dadurch wird die Aktivität zurückgesetzt und kann erneut gestartet werden. Damit die Daten aus dem OutContainer hierbei allerdings nicht verloren gehen, müssen sie mittels eines Loop-Connectors in den InContainer derselben Aktivität „zurückgeschleift" werden. Da diese zusätzlich notwendige Datenflusskante und die Exit-Condition das Prozessmodell deutlich unübersichtlicher machen, haben wir uns für die zuvor beschriebene Variante entschieden. Bei dieser ist die gesamten Algorithmik in dem einmalig erstellen, anwendungsunabhängigen Java-Code der Integrations-Infrastruktur konzentriert.

Beide Ansätze haben den Nachteil, dass ein Neustart der Aktivität im Log-File des WfMS protokolliert wird, was irrtümlicherweise als Problem- oder Fehlersituation interpretiert werden könnte. Idealerweise würde das verwendete WfMS das Schreiben von Anwendungsdaten auch dann erlauben, wenn die zugehörige Aktivität noch läuft. Dann wäre es problemlos möglich, (Zwischen-)Zustandsdaten im

eines InContainers zu lesen. Deshalb sind Daten aus dem InContainer sowohl für Nachfolgeaktivitäten wie auch für externe Programme (z.B. für Monitoring) voll nutzbar.

[2] Die API-Funktion „CheckOut" löst (im Gegensatz zu „Start") nicht tatsächlich das Starten einer Anwendung aus. Deshalb hat ihr (mehrfacher) Aufruf auch keine negativen Seiteneffekte. Es ist sogar schon beim ersten „CheckOut" (in Operation 3 aus Abb. 1) der Fall, dass die Aktivität in der Legacy Application bereits läuft (deren Start hat die Aktion schließlich angestoßen).

WfMS abzulegen. Da dies bei MQ Workflow aber nicht gegeben ist, wird der beschriebene „Work-around“ gewählt.

Für das Monitoring und die Visualisierung von Prozessen [3] ist es notwendig, auch Zugriff auf Anwendungsdaten der Applikation zu haben, und nicht nur auf die Prozess-Zustandsinformation. Zu diesem Zweck werden in Operation 2 Ergebnisdaten an die übergeordnete Prozessanwendung übergeben. Um diese Workflow-Applikation hierbei aber nicht zu überlasten, muss unterschieden werden, für welche Daten dies tatsächlich sinnvoll ist: Workflow-relevante Daten [10] müssen selbstverständlich übergeben werden, da sie für die Steuerung des Kontrollflusses vom WfMS benötigt werden. Kleinere Anwendungsdaten-Objekte können ebenfalls an das WfMS übergeben werden, wobei beachtet werden muss, dass bei einigen WfMS (u.a. auch MQ Workflow) die Anzahl der Elemente eines Daten-Containers beschränkt ist. Große Datenobjekte (z.B. CAD-Modelle, digitalisierte Bilder, Videosequenzen) sollten ausschließlich im Anwendungssystem gehalten werden, so dass dem WfMS lediglich ein Suchschlüssel übergeben wird. Zum Zwecke einer Prozessvisualisierung kann mit diesem dann auf die Anwendungsdaten zugegriffen werden.

4 Verwandte Arbeiten

In [8] wird dargelegt, dass es aktuell in der wissenschaftlichen Literatur kaum Ansätze gibt, die im Zuge einer Integration von Legacy Applications versuchen, deren Prozesswissen zu erschließen. Zwar wird in [4] eine prozessorientierte Kopplung realisiert, es werden aber keine Altanwendungen integriert. Stattdessen wird angenommen, dass die zu integrierende Anwendung selbst auf einem WfMS basiert. Dessen Kopplung wird notwendig, weil es zur Steuerung der übergeordneten Anwendung ungeeignet ist (z.B. da es sich um ein internes WfMS eines Produktdaten-Management-Systems handelt).

Weitere Ansätze, die noch entfernter mit der vorliegenden Fragestellung zu tun haben, werden in [8] ausführlich diskutiert. So untersucht z.B. [9], wie auf Komponententechnologie basierende Anwendungen in übergeordnete Workflows integriert werden können. Da allerdings nicht angenommen werden kann, dass jede Legacy Application diese Eigenschaft erfüllt, kann von diesem Ansatz im Allgemeinen nicht ausgegangen werden.

5 Zusammenfassung und Ausblick

In dem Beitrag wurde untersucht, wie eine prozessorientierte Legacy Application unter Erhaltung des Prozesswissens in eine Workflow-Anwendung integriert werden kann. Hierbei konnten alle identifizierten Anforderungen erfüllt werden, ohne bzgl. der Legacy Application irgendwelche Einschränkungen zu machen. Allerdings muss der Adapter, über den die Altanwendung angebunden ist, natürlich Zugriff auf diejenigen Daten und Ereignisse haben, die nach außen sichtbar gemacht werden sollen. Das vorgestellte Konzept ist insgesamt allgemeingültig, auch wenn der Ansatz für die

IBM-Produkte WebSphere MQ, WBI Message Broker und MQ Workflow detailliert und prototypisch implementiert wurde. Produkte mit ähnlicher Funktionalität sind aber auch von anderen Herstellern erhältlich.

Eine noch weitergehende Integration von Legacy Applications lässt sich erreichen, wenn zusätzlich zu deren Prozessmodell und Ausführungszustand auch noch weitere Modellinformationen integriert werden. Hierfür würde sich insbesondere das Organisationsmodell anbieten, da häufig sowohl Altanwendungen wie auch WfMS über ein solches verfügen. Werden diese (partiell) integriert, so reduziert sich der Aufwand für deren Pflege. Außerdem kann die Funktionalität der Prozesssteuerung erhöht werden, weil es durchaus die Anforderung gibt, dass die Bearbeiter einer Workflow-Aktivität von denen einer Legacy Application abhängig sind (z.B. selber Bearbeiter, Vorgesetzter, 4-Augen-Prinzip). Die Frage ist nun, wie solche Abhängigkeiten definiert und mit einer Ausführungssemantik versehen werden können, wenn die involvierten Systeme strukturell und inhaltlich unterschiedliche Organisationsmodelle aufweisen.

Denkt man die Idee einer Integrations-Infrastruktur noch weiter, so sollte sie nicht nur erlauben, Altanwendungen mit solchen prozessorientierten Anwendungssystemen zu integrieren, die auf einem WfMS basieren. Es sollte der Informationsaustausch zwischen beliebigen Anwendungssystemen unterstützt werden, auch zwischen zwei Legacy Applications. Dann übernimmt die Integrations-Infrastruktur die Rolle eines zentralen Verteilers, bei dem sich Anwendungen für gewisse Events registrieren können, der zur Run-Time die Zuordnung von Prozessinstanzen verwaltet, Informationen an (evtl. mehrere) Adapter und Zielsysteme weitergibt und dort ggf. die jeweils spezifizierten (ggf. unterschiedlichen) Operationen durchführt.

Literatur

1. O. Anspacher: Integration von heterogenen Anwendungen in Workflow-Management-Systeme. Diplomarbeit Fachhochschulen Ulm/Neu-Ulm. (2005)
2. T. Bauer: Integration heterogener Applikationstypen durch Workflow-Management-Technologie. In: Proc. Workshop on Enterprise Application Integration, Oldenburg. (2004) 39-47
3. T. Bauer: Visualisierung laufender Prozesse. In: Proc. Informatik 2004, Workshop Geschäftsprozessorientierte Architekturen, Ulm. (2004) 543-548
4. T. Bauer, T. Beuter, M. Weitner, T. Bär: Integration of Engineering and Production Planning Workflows. Journal of Advanced Manufacturing Systems 4(1). (2005) 37-52
5. IBM: MQSeries Application Programming Guide. (2000)
6. IBM: WebSphere MQ Workflow Programming Guide. Version 3.5. (2004)
7. S. Jablonski, M. Böhm, W. Schulze: Workflow-Management: Entwicklung von Anwendungen und Systemen. dpunkt-Verlag. (1997)
8. R. Pryss: Enterprise Application Integration: Anforderungen, Ansätze und Technologien. Diplomarbeit Universität Ulm. (2005)
9. S. Schreyjak: Coupling of Workflow and Component-Oriented Systems. Lecture Notes in Computer Science, 1357. (1998)
10. Workflow Management Coalition: Terminology & Glossary. Document Number WFMC-TC-1011. (1999)

Integration und Orchestrierung von Geschäftsprozessen in Web Applikationen – Ein Service-Orientierter Ansatz

Jochen Müller, Andreas Kümpel, Paul Müller

AG Integrierte Kommunikationssysteme
Universität Kaiserslautern
Gottlieb-Daimler-Straße
67663 Kaiserslautern
{jmueller, a_kuempel, mueller}@informatik.uni-kl.de

Abstract: Mit dem zunehmenden Einsatz des Electronic Business werden Geschäftsprozesse und Anwendungssysteme über die Grenzen eines Unternehmens geöffnet und mit denen anderer Geschäftspartner kombiniert. Die „Business Process Execution Language for Web Services“ ermöglicht es, Geschäftsprozesse zu beschreiben und einzelne Web Services miteinander zu kombinieren, um durchgängige Geschäftsprozesse sowohl auf Anbieter, als auch auf Kundenseite zu unterstützen. In dieser Arbeit wird ein Konzept vorgestellt wie existierende Web Portale erweitert werden, um so die Abarbeitung von Geschäftsprozessen service-orientiert zu ermöglichen.

1 Einleitung

Das Electronic Business führt zu einer erheblichen Umgestaltung von Markt- und Wertschöpfungsstrukturen. Die verteilte Struktur und die große Verfügbarkeit des Internets erlauben eine Rekonfiguration von Wertschöpfungsketten. Geschäftsprozesse und Anwendungssysteme werden über die Grenzen eines Unternehmens geöffnet und mit denen anderer Partner kombiniert [ZWBL01].

Die im Laufe der Zeit in Unternehmen typischerweise entstandene Applikationslandschaft ist dadurch gekennzeichnet, dass häufig heterogene und spezialisierte Informationssysteme für die einzelnen Bereiche eines Unternehmens (Finanzbuchhaltung, Materialwirtschaft, etc.) entwickelt wurden, mit den bekannten Schwachstellen wie Datenredundanz oder Probleme in der Wartbarkeit. Durch so genannte Querschnittsysteme wird seit den 90ger Jahren versucht diese abzufangen, und die Systeme miteinander verbinden. Portale lassen sich zu diesen Querschnittsystemen zählen. Dabei handelt es sich um Anwendungen, die das Internet als Kommunikationsplattform erschließen und sich stark an Geschäftsprozesse ausrichten. Ein Portal kann als eine Web-Anwendung definiert werden, welche Inhalte, Dienste und Funktionen integriert [MHM04].

Der Trend beispielsweise bei Kunden- oder Mitarbeiterportalen geht dazu hin, die offerierten Dienste mit existierenden Geschäftsprozessen, zugrunde liegenden Unternehmensinformationssystemen und Anwendungen zu integrieren.

2 Service-Orientierte Architekturen

Service-Orientierte Architekturen (SOA) sind gekennzeichnet durch verteilte Komponenten, deren Daten und Funktionalität mittels *Services (Dienste)* bereitgestellt werden. Ein Service kapselt die interne Struktur und bietet nach außen eine klar definierte Schnittstelle. Services können dezentral bereitgestellt und über das Netzwerk angesprochen werden. Eine entscheidende Eigenschaft von SOA ist die lose Koppelung. Darunter versteht man dass keine Festlegung bzgl. konkreter Implementierung und eingesetztem Kommunikationsprotokoll getroffen wird [CuDu02].

Bei einer SOA lassen sich drei unterschiedliche Rollen definieren: Der Dienst-Anbieter, der Dienst-Konsument und der Dienst-Vermittler. Realisierungen und Implementierungen basieren fast ausschließlich auf dem Web Service Ansatz. Zu weiteren Ausführungen über SOA wird verwiesen auf [CuDu02].

Die Bedeutung und die Verbreitung von Web Services haben in den vergangenen Jahren erheblich zugenommen. Im Zuge der immer stärkeren Vernetzung von Rechnern steigt der Bedarf an lose gekoppelten und verteilten Anwendungen. Seit kurzer Zeit ist die Verbindung von Web Services mit Orchestrierungssprachen zu Workflows ein aktives Forschungsthema.

BPEL4WS (auch BPEL) ist die Abkürzung für Business Process Execution Language for Web Services. BPEL4WS ist eine XML Sprache mit der Geschäftsprozesse spezifiziert werden können. BPEL ermöglicht es, verteilte Web Services zu Geschäftsprozessen zusammenzufassen und unterstützt dabei die Interaktion zwischen den Web Services untereinander und die Interaktion mit dem Client. Auf diese Weise ist es möglich, die Geschäftslogik des Prozesses von der eigentlichen Anwendung zu trennen [Cu03].

BPEL ist mittlerweile zu einem anerkannten Standard für die Komposition von Web Services geworden. Mit BPEL ist es möglich komplexe Prozesse zu erstellen, indem unterschiedliche Aktivitäten einzeln realisiert und anschließend zusammengefasst werden können [Aals03]. Bei den Aktivitäten kann es sich beispielsweise um den Aufruf von Web Services handeln, die Manipulation von Daten oder das Abfangen und Behandeln von Fehlern innerhalb des Prozesses. Diese einzelnen Aktivitäten können verschachtelt werden und zu Abläufen, wie Schleifen oder einer parallelen Abarbeitung, kombiniert werden [Cu03].

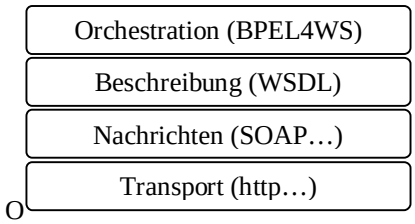

Abbildung 1: BPEL und Web Service Protokoll Hierarchie [Cu03]

Abbildung 3 zeigt die Position von BPEL in der Web Service Architektur. Über die unterste Schicht, der Transport-Schicht tauschen die Web Services bzw. der BPEL-Geschäftsprozess Informationen aus. Die versendeten Nachrichten werden über Protokolle wie HTTP (Hypertext Transfer Protocol), SMTP (Simple Mail Transfer Protocol) oder FTP (File Transfer Protocol) übertragen. In der darüber liegenden Nachrichten-Schicht werden die übertragenen Informationen de- bzw. encodiert. Verwendet werden hier auf XML basierende Formate wie XML-RPC oder SOAP. Die dritte Schicht realisiert die Beschreibung des Web Services, d.h. seine Schnittstelle nach außen. Zur Beschreibung wird sowohl für Web Services als auch für BPEL-Geschäftsprozesse WSDL (Web Service Description Language) verwendet. In der obersten Schicht, der Orchestrierungs-Schicht erfolgt die Beschreibung, wie die Web Services miteinander interagieren können. Hier wird die Business Logik und die Ausführungsreihenfolge der Web Services beschrieben. Die Orchestrierung kann beispielsweise mit BPEL4WS erfolgen.

Bei Geschäftsprozessen in BPEL wird grundsätzlich zwischen *Ausführbaren Geschäftsprozessen* und *Abstrakten Geschäftsprozessen* [Cu03] unterschieden. Ausführbare Geschäftsprozesse beschreiben den kompletten Ablauf eines Prozesses sowie dessen Interaktion mit anderen Web Services. Mit BPEL4WS beschriebene ausführbare Geschäftsprozesse können von einer BPEL4WS-Engine gelesen, instanziiert und ausgeführt werden. Abstrakte Geschäftsprozesse hingegen beschreiben nur einen Teil eines Prozesses. So werden die öffentlichen Rahmenbedingungen beim Nachrichtenaustausch festgelegt und der Prozess allgemein beschrieben. Diese abstrakte Beschreibung dient dazu, die Integration und das Verständnis des Prozessablaufes für die Partner zu vereinfachen, die mit dem Prozess interagieren möchten. Interne, eventuell auch vertrauliche Details des Prozesses bleiben dem Partner dabei verborgen.

Im Folgenden wird auf ausführbare Geschäftsprozessbeschreibungen mit BPEL fokussiert.

3 Web Service Integration Architektur (WESIR)

Die meisten Webbasierten Systeme bauen auf einem drei Ebenenmodell (3-Tier-Architektur) auf. Die erste Ebene, das Frontend ermöglicht die Interaktion mit dem Nutzer. Die dritte Ebene, das Backend oder auch Enterprise Information System Tier, ist für die Datenspeicherung verantwortlich. Die Mittel-Schicht abstrahiert die beiden Schichten und ermöglicht so eine effiziente Kommunikation zwischen Front- und Backend.

Zur Modellierung des Middle-Tiers hat sich das Model View Controller (MVC) Design Pattern etabliert. Dieses Pattern entkoppelt die Anwendungs-, Darstellungs- und Interaktionslogik und ermöglicht so einfache nachträgliche Änderungen und Erweiterungen. Vor der Verwendung des MVC-Patterns, war es üblich, den Programmcode für die Geschäftslogik, die Datenhaltung und –manipulation sowie die Benutzeroberfläche vollkommen miteinander zu verflechten [HiRe03]. Diese Verflechtungen haben zur Folge, dass die Wartbarkeit dieser Applikationen stark erschwert wird. Eventuell notwendige Erweiterungen in der Applikation können so zu einem enormen Arbeitsaufwand führen, da oftmals redundante Implementationen derselben Prozeduren in unterschiedlichen Teilen des Programms notwendig wird. Auch wird die Möglichkeit der Widerverwendbarkeit von Komponenten stark verringert.

Aus diesen Gründen wurden die Komponenten für einen Service-Orientierten Ansatz an MVC angepasst. Die angepasste Architektur ist in Abbildung 1 dargestellt. Ziel ist es das die verschiedenen Prozessschritte dem Akteur über ein einheitliches Interface dargestellt werden. Dieser „Master Service“ setzt sich aus verschiedenen einzelnen Web Service zusammen. Die Erweiterungen der Architektur beziehen sich folglich nur auf den Controller. Dieser muss um Komponenten erweitert werden, welche es ermöglichen Web Service als auch orchestrierte Service zu verarbeiten.

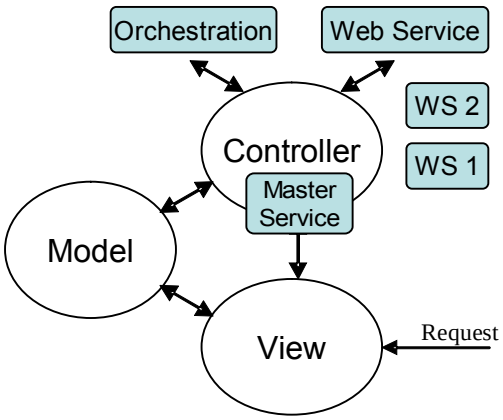

Abbildung 1: Architektur von SESAME

4 Fallstudie zur Flächennutzungsplanung

4.1 Hintergründe zur Flächennutzungsplanung

In einem Flächennutzungsplan (FNP), auch vorbereiteter Bauleitplan genannt, wird festgelegt, wie die Flächen innerhalb einer Gemeinde genutzt werden sollen. Der Flächennutzungsplan soll die verschiedenen räumlichen Nutzungsansprüche, wie beispielsweise Wohnen, Verkehr, Gewerbe oder Grünflächen koordinieren und zu einem abgewogenen Gesamtkonzept zusammenzuführen. Die Flächennutzungsplanung befasst sich mit der Erstellung dieser vorbereiteten Bauleitpläne.

Die Praxis der Flächennutzungsplanung ist jedoch sehr komplex und insbesondere zeitintensiv. Bis zu einer endgültigen Genehmigung eines neuen Flächennutzungsplanes können oftmals mehrere Jahre vergehen. Zu diesem Zeitpunkt kann der beschlossene FNP schon längst an Aktualität verloren haben, so dass eventuell ein neuer Plan erstellt werden muss [Stei03].

Beider Flächennutzungsplanung handelt es sich um einen typischen verteilten Geschäftsprozess der durch ein komplexes rechtliches Rahmenwerk vorgegeben wird, welches die Partizipation unterschiedlicher Akteure vorsieht, und unterschiedliche Studien (soziologische, ökologische...) als Datenbasis vorschreibt. Im Einsatz moderner Informations- und Kommunikationstechnologien zur Unterstützung des gesamten Workflows wird ein hohes zeitliches Optimierungspotential gesehen [MHM04]. In einem ersten Schritt wurde ein Web Portal aufgebaut welches die Informationen prozessorientiert bündelt und bereitstellt [HiRe03]. Ziel ist es die derzeit durchschnittliche Realisierungszeit von ca. 8 Jahren auf 4 Jahre zu verkürzen.

4.2 Anwendungsfall: Beteiligung Träger öffentlicher Belange

Den zentralen Punkt im Rahmen des Flächennutzungsplanprozesses stellt eine Verteilung der relevanten Informationen (bspw. aktuelle, regionale geographische und demographische Daten) für die Prozessbeteiligten dar. Diese prozessorientierte Bereitstellung der benötigten Daten und Informationen, kann in eine Web-Architektur abgebildet werden.

Darüber hinaus erscheint auch eine Unterstützung des Workflows zur Erstellung eines Flächennutzungsplanes notwendig und sinnvoll. Die bei der Erstellung beteiligten Akteure und Instanzen wie etwa Gemeinden, Träger öffentlicher Belange (TöB) oder die Bürger, müssen sich an den gesetzlich geregelten Ablauf halten sowie zeitliche Fristen einhalten. Eine Modellierung dieser Vorgaben im Rahmen der Architektur erscheint sinnvoll und ist für die Gemeinden von großem Mehrwert.

Um die Geschäftsprozesse weiter zu unterstützen und zu automatisieren soll das Portal mit dem oben beschriebenen Konzept WESIR erweitert werden. Ziel ist es die einzelnen Prozesse als Web Service zu realisieren und diese anschließend zu orchestrieren, so dass eine Zusammenspiel der Akteure unterstützt wird.

Als Anwendungsfall wurde der Prozess „Beteiligung Träger öffentlicher Belange“ im Rahmen der Flächennutzungsplanung untersucht. Ein Teilprozess wird in Abbildung 2 als Sequenzdiagramm dargestellt.

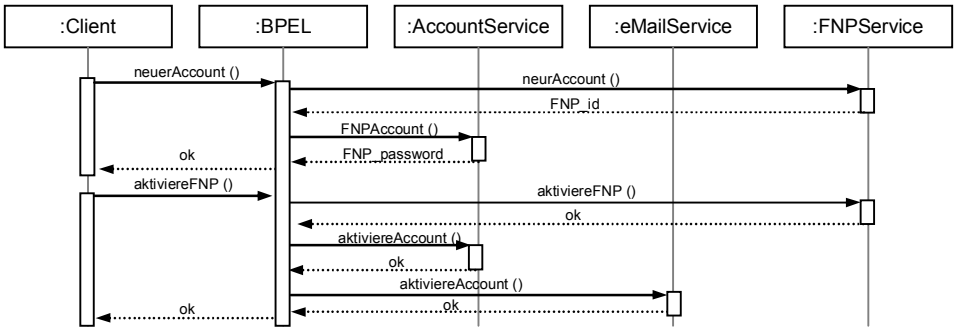

Abbildung 2: Sequenzdiagramm Beteiligung der TöB

Im ersten Schritt legt der Client einen neuen Flächennutzungsplan an. Dabei werden von ihm alle notwendigen Daten übermittelt, inklusive eine Liste aller beteiligten TöBs und deren E-Mail-Adressen. Der BPEL-Prozess ruft weitere Web Services auf, um den FNP in der Datenbank anzulegen und die TöB-Accounts einzurichten. In einem zweiten Schritt erfolgt die eigentliche Aktivierung des FNP, d. h. die Einsichtnahme in die Daten wird sowohl der Öffentlichkeit, als auch den TöB ermöglicht. Die TöB werden zusätzlich per E-Mail darüber informiert, dass ein neuer FNP vorliegt. Die benötigten Zugangsdaten, um eine Stellungnahme über das Web-Frontend zu verfassen, werden dabei mitgeschickt.

Der BPEL-Prozess wartet nun solange, bis das Ablaufdatum der Einsichtnahme erreicht ist. In dieser Zeit erfolgt keine weitere Interaktion mit dem Client. Parallel können in diesem Zeitraum die TöB Stellungnahmen und Anträge auf Verlängerung verfassen. Dies geschieht jedoch unabhängig vom BPEL-Prozess. Nach Ablauf der Einsichtnahme wird der Status des FNP entsprechend gesetzt und die TÖB-Accounts deaktiviert.

Zur Abbildung dieses workflows wurden vier Web Service identifiziert, welche während dieses Prozesses benötigt werden.

Der *FNPManager* beinhaltet Funktionen zum Anlegen und Ändern eines Flächennutzungsplanes. Neben Möglichkeiten wie der Statusänderung eines FNP, kann über den Web Service überprüft werden, ob das Ablaufdatum der Einsichtnahme eines FNP überschritten wurde. Der *FNPAccountManager* ermöglicht das Anlegen, sowie Aktivieren und Deaktivieren von TÖB-Accounts. Zusätzlich kann durch das Web-Frontend über diesen Web Service überprüft werden, ob die Zugangsdaten eines TÖB korrekt sind.

Der *FNPEmailService* bietet die Möglichkeit, vorgefertigte E-Mails zu versenden. Soll eine Benachrichtigung an alle beteiligten TÖB über einen neuen Flächennutzungsplan verschickt werden, so brauchen an den Web Service nur Informationen wie Titel und Beschreibung des FNP sowie die E-Mail-Adressen und Zugangsdaten der TÖB übermittelt werden. Der *FNPAttachmentService* bietet die Möglichkeit, binäre Daten wie Anhänge zu Flächennutzungsplänen oder Stellungnahmen in der Datenbank zu speichern und aus der Datenbank wieder auszulesen. Dieser Service wird nur direkt vom Web-Frontend angesprochen.

4.3 Realisierungsansatz

Es wird nun beschrieben wie der in Kapitel 3 vorgestellte Ansatz in eine bestehende Referenzimplementation integriert wurde. Die Realisierung wurde auf das in [HiRe04] vorgestellte Portal-Framework angewandt. Zentrale Elemente sind die PostgreSQL Datenbank zur Datenhaltung, JavaBeans zur Implementierung der Geschäftslogik und der Tomcat Servlet Container zur Verarbeitung der dynamischen Seiten.

Für die Realisierung war es zunächst notwendig die Architektur so zu erweitern dass sowohl Web Service, als auch orchestrierte Service ausgeführt werden können. Zur Ausführung der Web Service mit dem Tomcat-Server wurde auf Axis zurückgegriffen. Darüber hinaus wurde die BPWS4J-Engine von IBM integriert [IBM04]. Die Architektur ist in Abbildung 3 dargestellt.

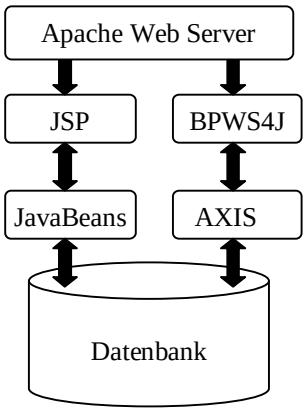

Abbildung 2: Komponentenübersicht der Architektur

5 Zusammenfassung

Die Einbettung von Geschäftsprozessen in bestehende IT-Infrastrukturen, wie Informationsportale, wird in Zukunft für Organisationen eine immer größere Rolle spielen. Die „Business Process Execution Language for Web Services“ bietet einen sehr guten Ansatz, um eine plattformunabhängige Webdienstkommunikation zur Koordination und Abarbeitung solcher Geschäftsprozesse zu ermöglichen.

Das hier vorgestellte Konzept und der entwickelte Prototyp können als Grundstein angesehen werden, um verteilte Prozesse zu integrieren. Speziell im Bereich der Flächennutzungsplanung zeigt der Prototyp, dass eine computerunterstützte Planungsabwicklung zu erheblichen Zeit- und Kosteneinsparungen führen kann, insbesondere durch die schnelle Informationsbeschaffung und –bearbeitung. Durch die Verwendung von Web Services und BPEL4WS besteht weiterhin die Möglichkeit, Teile des Fontends beispielsweise in Mitarbeiterportale oder anderen Anwendung im Bereich der Flächennutzungsplanung relativ einfach zu integrieren. Zurzeit wird der Prototyp noch einer genaueren Prüfung unterzogen. Ziel ist es demnächst mit Rheinland-Pfälzischen Gemeinden die Praxistauglichkeit des Systems zu testen.

Literaturverzeichnis

[Aals03] W.M.P. Van der Aalst: Don´t go with the flow: Web services composition standards exposed aus: IEEE Intelligent Systems (18), Jan/Feb 2003, 72-76 Januar 2003

[Cu03] Francisco Curbera et al.: BPEL/BPEL4WS - Business Process Execution Language for Web Services, Version 1.1, „http://www.ibm.com/developerworks/ library/ws-bpel/“, 2003

[CuDu02] F. Curbera, M. Duftler et al: Unraveling the Web Services Web, IEEE Internet, Computing, March/April 2002, 86-93, 2002

[IBM04] IBM alphaWorks, *BPWS4J*, http://www.alphaworks.ibm.com/tech/bpws4j

[HiRe03] M. Hillenbrand and B. Reuther, “Building Block for Web Applications”, 7th IASTED IMSA Hawaii, USA, 8/2003, pp. 757-761.

[MHM04] J. Müller, D. Henrici, P. Müller: Computer-Aided Dynamic Processes for Urban Land Use Planning , in Work in Progress Session of 30th EUROMICRO, Rennes, France, 2004

[MHH05] J. Müller, D. Henrici, M. Hillenbrand: Peer-to-Peer Architekturen für verteilte Geschäftsprozesse: Überlegungen zur Flächennutzungsplanung, Workshop "Peer-to-Peer-Systeme und -Anwendungen", 14. Fachtagung Kommunikation in Verteilten Systemen 2005

[Stei03] G. Steinebach: Informations- und Kommunikationssysteme im Verfahren der Bauleit-planung, in: Zeitschrift für deutsches und internationales Bau- und Vergaberecht, 1/2004, S.16

[ZWBL01] C. Zirpins, H. Weinreich, A. Bartelt, and W. Lamersdorf, “Advanced Concepts for Next Generation Portals”, 12th International Workshop on Database and Expert Systems Applications, IEEE Computer Society, Los Alamos, USA, 9/2001, pp. 501-506.

Anforderungen an Informationsobjekttypen als Basis von Architekturentscheidungen bei der Datenintegration

Reinhard Jung

Universität Bern, Institut für Wirtschaftsinformatik,
Engehaldenstrasse 8, 3012 Bern, Schweiz,
reinhard.jung@iwi.unibe.ch

Zusammenfassung. Für die Datenintegration sind inzwischen viele Architekturkonzepte und Technologien verfügbar, und es stellt sich die Frage, auf welcher Basis eine Architektur- und Technologieauswahl erfolgen sollte. Der Beitrag stellt aus fachlicher Perspektive (Sicht der Aufgabenträger) eine Reihe von nicht-funktionalen und funktionalen Anforderungen im Zusammenhang mit Informationsobjekttypen vor, aus denen sich potenziell technische Anforderungen an Datenintegrationsarchitekturen ableiten lassen.

1 Problemstellung

Der Wandel von Geschäftsmodellen und eine immer dynamischer werdende Produktpolitik zwingt die Unternehmen, in immer kürzeren Zeitabständen neue Softwarelösungen in Betrieb zu nehmen. In analoger Form gilt diese Feststellung auch für Behörden, da sich auch ihr Leistungsportfolio bei sinkenden Budgets ständig erweitert oder zumindest stark verändert.

Aus der geschilderten Situation ergibt sich die Herausforderung, neue Anwendungssysteme mit den bestehenden integrieren zu müssen, um die Effektivität des betrieblichen Informationssystems aufrecht zu erhalten oder herzustellen. Ein zentraler Schritt eines solchen Integrationsvorhabens ist die Integration der Daten aller beteiligten Systeme (Datenintegration).

Prinzipiell stehen eine ganze Reihe von Architekturkonzepten zur Verfügung, die für eine Integration grundsätzlich geeignet sind. Beispiele sind so unterschiedliche Konzepte wie Data-Warehouse-Systeme, Operational-Data-Stores und serviceorientierte Architekturen. Darüber hinaus ist auch im Bereich der Technologien inzwischen eine grosse Menge an Optionen verfügbar, beispielsweise Middleware-Produkte und -Standards, Integrations- oder Message-Broker und Web Services. Bevor eine konkrete Architektur gestaltet und eine Entscheidung für bestimmte Technologien und Software-Lösungen gefällt werden kann, muss allerdings eine genaue Spezifikation der Anforderungen erfolgen, denn viele Architekturkonzepte sind nicht universell einsetzbar. Ein Data-Warehouse-System beispielsweise ist – zumindest in der „klassischen“ Sichtweise dieses Architekturkonzepts (eine grundsätzlich andere Sichtweise wird vertreten bspw. in [15]) – aufgrund der Charakteristika von Datenextraktions- und Konsolidierungsprozessen nicht in der Lage, Echtzeit-Daten bereit-

zustellen oder etwa die Modifikation von Data-Warehouse-Daten in die operativen Anwendungssysteme zu propagieren. Der vorliegende Beitrag widmet sich der Frage, wie sich Anforderungen an Architekturen für die Datenintegration aus fachlicher Sicht und damit aus Sicht des Informationsbedarfs beschreiben lassen. Ferner wird diskutiert, inwieweit sich diese Merkmale potenziell auf die Auswahl einer technisch geeigneten Architektur für die Datenintegration auswirken.

2 Konzeptioneller Rahmen und Begriffe

Als Basis der weiteren Ausführungen wird hier zunächst ein konzeptioneller Rahmen beschrieben, der im Wesentlichen aus einer Basishypothese und einigen zentralen Begriffsdefinitionen besteht.

Für Zwecke des vorliegenden Beitrags ist zunächst festzulegen, worin das (Entwicklungs-)Ziel liegt: Es sei gegeben durch die Implementierung einer Datenintegrationsarchitektur. Der Zweck dieser Architektur liegt darin, den Aufgabenträgern innerhalb eines Untersuchungsbereichs einen effizienten Zugriff auf die Daten zu ermöglichen, die sie für ihre Aufgabenerfüllung benötigen. Dabei ist davon auszugehen (Basishypothese), dass ein effizienter Zugriff auf diese Daten mit Hilfe der vorhandenen Anwendungssysteme („legacy“) bzw. der vorhandenen Anwendungsarchitektur nicht möglich ist. Da eine vollständige Ersetzung der Anwendungsarchitektur aus Gründen des Investitionsschutzes nicht realistisch erscheint, wird statt dessen angestrebt, die Anwendungsarchitektur mit Blick auf den genannten Zweck zu ergänzen. Die Ergänzung besteht in so genannten Integrationskomponenten, die zusammen mit der Anwendungsarchitektur die Datenintegrationsarchitektur bilden (vgl. **Abb. 1**).

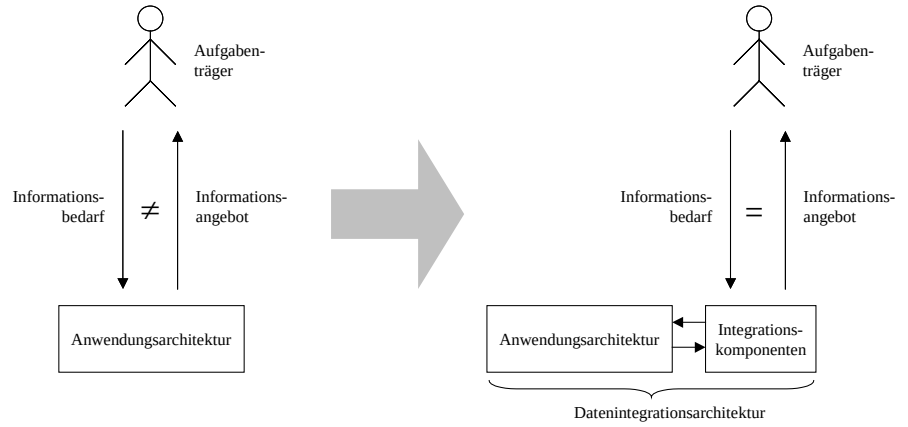

Abb. 1. Ausgangssituation beim Entwurf einer Datenintegrationsarchitektur

Zu konkretisieren ist der in der Abbildung verwendete Begriff „Information“. Die computergestützte Verarbeitung von Daten basiert auf Datenelementen (z.B. „Meier“, „400“) oder allgemein auf Datenelementtypen (z.B. Kundenname, Bestellmenge). Aus Sicht einer betrieblichen Transaktion sind die Datenelementtypen zu Datenobjekttypen zusammengefasst (z.B. Kunde, Bestellung), wobei Datenobjekttypen zuein-

ander in Beziehung stehen können; in Entity-Relationship-Modellen entspricht dieser Zusammenhang den Beziehungstypen, die Entitätstypen miteinander verbinden.

Die Sicht eines Aufgabenträgers orientiert sich nicht an der Menge der vorhandenen Datenelemente, sondern an dem Bedarf, der sich aus der Aufgabenausführung für ihn subjektiv ergibt. Im Vordergrund stehen also zweckorientierte Daten, die man entsprechend als Information bezeichnen kann. Auch die an Transaktionen orientierte Zusammenfassung der Daten entspricht nicht der Sichtweise eines Aufgabenträgers. Ein Beispiel ist die Analyse einer Verkaufsstatistik, in der miteinander in Beziehung stehende Ausprägungen von Informationselementtypen (z.B. Region, Verkaufsmenge, Verkaufspreis, Kundenname) enthalten sind. Ein verallgemeinertes Konzept, das als Ganzes den Gegenstand einer Aufgabe darstellt, wird im Folgenden als Informationsobjekttyp bezeichnet (z.B. eine Verkaufsstatistik). In **Abb. 2** ist der Zusammenhang zwischen den Konzepten der fachlichen Sicht (Informationsobjekttyp und Informationselementtyp) und der technischen Sicht (Datenobjekttyp und Datenelementtyp) dargestellt. Die Notation ist so zu verstehen, dass beispielsweise ein Informationsobjekttyp aus mindestens zwei Informationselementtypen besteht, während ein Informationselementtyp in mindestens einem Informationsobjekttypen enthalten ist, aber auch in mehreren enthalten sein kann.

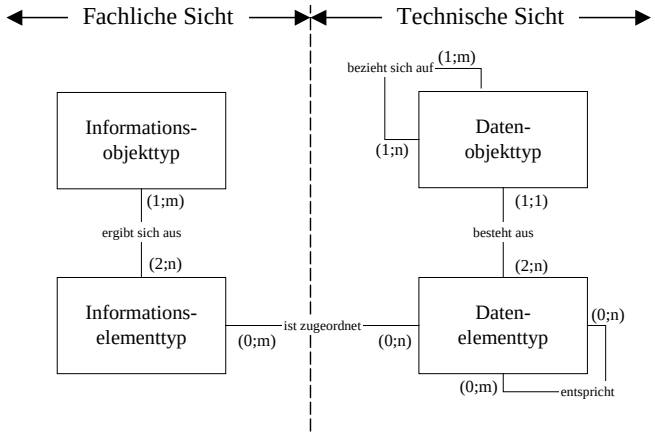

Abb. 2. Zusammenhang zwischen Informationsobjekt-/-elementtypen und Datenobjekt-/-elementtypen

Orientiert man sich an dem vorgestellten konzeptionellen Rahmen, so werden die Anforderungen eines Aufgabenträgers (fachliche Sicht) bezogen auf Informationsobjekte und -elemente oder allgemeiner auf der entsprechenden Typebene festgelegt. Aus technischer Sicht ist später zu prüfen, ob diese Anforderungen auf Basis der verfügbaren Datenelemente und -objekte erfüllbar sind bzw. welche Art von Datenintegrationsarchitektur zu realisieren ist.

3 Merkmale von Informationsobjekttypen und -elementtypen

Bereits eine Reihe von Autoren hat sich der Frage gewidmet, wie der Informationsbedarf eines Aufgabenträgers ermittelt werden kann, also wie die relevanten Informationsobjekttypen und -elementtypen zu erheben sind (vgl. z.B. [4], [10], [11], [13]). Als Ergänzung zu diesem inhaltlichen Aspekt von Informationsbedarfsanalysen wird hier die Fragestellung aufgeworfen, welche Merkmale der Informationsobjekt- und Informationselementtypen spezifiziert werden sollten. Als Anforderungen im Sinne dieses Beitrags werden (vom Aufgabenträger) gewünschte Ausprägungen dieser Merkmale aufgefasst.

In Anlehnung an das Requirements Engineering kann eine Klassifikation vorgenommen werden (siehe **Abb. 3**) einerseits in nicht-funktionale Merkmale, deren Ausprägungen die gewünschten Eigenschaften der Informationsobjekttypen und –elementtypen unabhängig von ihrer Verwendung (statische Sicht) repräsentieren. Eine zweite Klasse bilden andererseits die funktionalen Merkmale, deren Ausprägungen die gewünschten Eigenschaften der Informationsobjekttypen und -elementtypen bei der Verwendung (dynamische Sicht) abbilden.

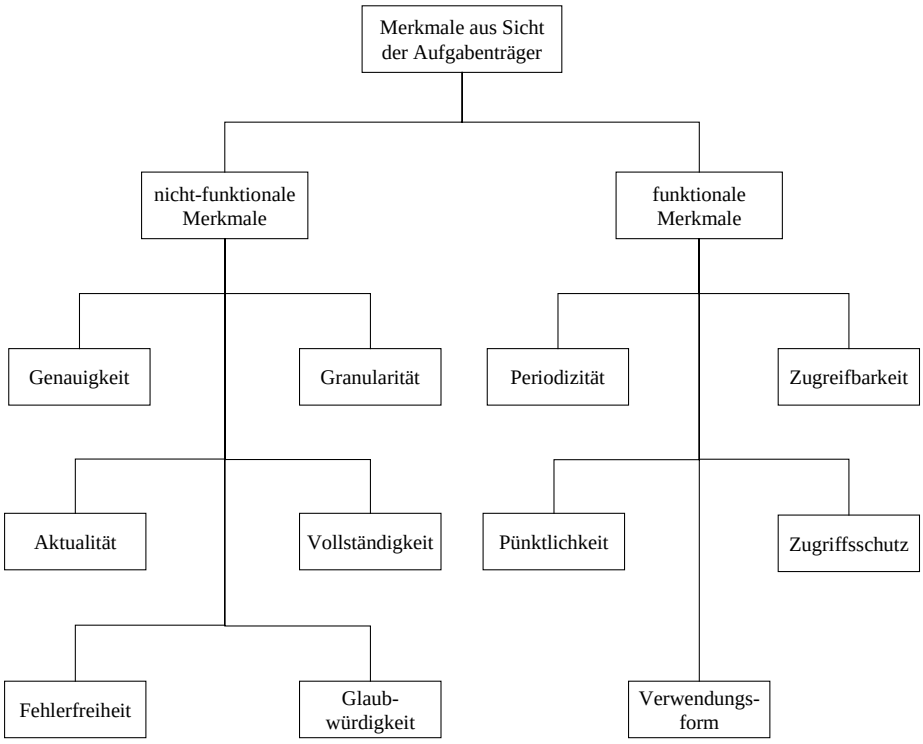

Abb. 3. Merkmale von Informationsobjekttypen und -elementtypen aus Sicht der Aufgabenträger

Die Herleitung der Merkmalsmenge kann hier nicht vollständig wiedergegeben werden (zu Details vgl. [5]). Sie basiert auf einer Recherche in der Literatur zur Daten-

qualität (vgl. [12], [14]) und in der betriebswirtschaftlichen Literatur (vgl. [1], [3], [6]). Bei den Merkmalen wird jeweils durch einen Buchstaben markiert, ob sich das jeweilige Merkmal auf Informationsobjekttypen („O“) und/oder -elementtypen („E“) bezieht.

3.1 Qualitative Anforderungen (nicht-funktionale Anforderungen)

Genauigkeit (O): Die Genauigkeit gibt an, wie detailliert Realweltobjekte durch Informationselemente beschrieben werden (vgl. [8] und auch [2]). Als Operationalisierung dieser Definition kann einerseits die Anzahl der für die Beschreibung verwendeten Informationselementtypen genutzt werden und andererseits auch das Skalenniveau dieser Typen; je höher das Skalenniveau desto grösser die Genauigkeit.

Aktualität (O/E): Die Aktualität drückt aus, inwieweit ein Informationselement die tatsächliche gegenwärtige Eigenschaft des zugrunde liegenden Realweltobjekts beschreibt.

Fehlerfreiheit (O/E): Ein Informationsobjekt ist bezogen auf einen bestimmten Zeitpunkt fehlerfrei, wenn die in ihm enthaltenen Informationselemente die Eigenschaften der zugehörigen Realweltobjekte in jenem Zeitpunkt widerspiegeln.

Glaubwürdigkeit (O/E): Unter der Glaubwürdigkeit lässt sich die Einschätzung subsumieren, ob bestimmte Informationsobjekte hinsichtlich ihrer übrigen Qualitätsmerkmale als geeignet eingestuft werden, ohne dass der betreffende Aufgabenträger für diese Merkmale die genauen Ausprägungen kennt.

Granularität (E): Die Granularität eines Informationselements drückt aus, ob und wie viele Einzelbeobachtungen in der Realwelt zusammengefasst wurden; in [2] wird für dieses Merkmal die Bezeichnung „Verdichtung“ verwendet. Von einer groben Granularität wäre beispielsweise im Fall eines Informationselementtyps „Jahresumsatz“ zu sprechen, der sich aus der Summierung von Einzelumsätzen ergibt.

Vollständigkeit (O): Die Vollständigkeit gibt an, inwieweit ein Informationsobjekt alle geforderten Informationselemente enthält. Im oben genannten Beispiel der Verkaufsstatistik wären fehlende Verkaufsmengen für bestimmte Regionen ein Indiz für Unvollständigkeit.

3.2 Funktionale Anforderungen

Periodizität (O): Im betrieblichen Berichtswesen werden drei Berichtsformen unterschieden, nämlich Standard-, Abweichungs- und Bedarfsberichte (vgl. [3]). Standardberichte sind regelmässig zu erstellen, und für Abweichungsberichte müssen in einem vorgegebenen Rhythmus Ist- mit Soll-Werten verglichen werden, sodass auch hier eine Regelmässigkeit zugrunde liegt. Bedarfsberichte werden hingegen in unregelmässigen Abständen von Aufgabenträgern angefordert. Auf die vorliegende Fragestellung übertragen, drückt die Periodizität aus, in welchem Rhythmus Informationsobjekte und -elemente von einem Aufgabenträger benötigt werden.

Pünktlichkeit (O): Die Pünktlichkeit setzt das Eintreffen eines Informationsobjekts beim Aufgabenträger (Zeitpunkt t_1) mit dem Zeitpunkt in Beziehung, zu dem das Eintreffen erwartet wurde (t_2). Liegt t_1 nach t_2, dann ist von einer unpünktlichen Bereit-

stellung zu sprechen. Die Pünktlichkeit steht in enger Beziehung zur Antwortzeit bei der Anforderung eines Informationsobjekts, deren Länge die Pünktlichkeit massgeblich beeinflusst. Die Pünktlichkeit wurde als Merkmal vorgezogen, weil sie aus Sicht des Aufgabenträgers das wichtigere Merkmal ist.

Verwendungsform (O/E): Die Verwendungsform gibt an, welche Operationen auf den Informationsobjekten vorgesehen sind. Eine Verkaufsstatistik wird typischerweise für rein informative Zwecke angefordert (Verwendungsform „lesend“). Es sind darüber hinaus durchaus Szenarien denkbar, in denen Informationsobjekte auch modifiziert werden müssen (Verwendungsform „lesend/schreibend“), beispielsweise Budgetierungsprozesse.

Zugreifbarkeit (O): Die Zugreifbarkeit drückt aus, ob der Aufgabenträger die (vorhandenen) Informationsobjekte auch tatsächlich erhalten kann. Bei diesem Merkmal besteht ein enger Zusammenhang zur Pünktlichkeit; eine fehlende oder eingeschränkte Zugreifbarkeit führt unweigerlich zur Unpünktlichkeit.

Zugriffsschutz (O): Integrierte Daten haben häufig sensitiven Charakter. Insofern ist auch festzulegen, inwieweit sie vor unberechtigtem Zugriff zu schützen sind. Dieses Merkmal ist nicht nur im Sinne von Anforderungen aus Sicht des Aufgabenträgers festzulegen, sondern gegebenenfalls auch aus Sicht der Organisation, denn unter Umständen können oder dürfen nicht jedem Aufgabenträger die gewünschten Informationsobjekte zugänglich gemacht werden.

4 Implikationen aus technischer Sicht

Geht man davon aus, dass ein Aufgabenträger seinen Informationsbedarf spezifiziert und jeweils angegeben hat, welche Ausprägungen die qualitativen Merkmale der Informationsobjekte und -elemente jeweils aufweisen sollten, dann stellt sich die Frage, welche Konsequenzen sich daraus aus technischer Sicht ergeben. Grundsätzlich ist dabei zu berücksichtigen, dass

- der Informationsbedarf durch Integration mehrerer Datenquellen zu decken ist, nämlich der verschiedenen Datenquellen innerhalb der vorhandenen Anwendungsarchitektur (Anwendungs- und Datenverwaltungssysteme);
- diese Datenquellen möglicherweise autonom sind und daher in ihrem Antwortverhalten nur begrenzt oder gar nicht beeinflusst werden können.
- zwischen den Datenquellen bezogen auf einzelne Datenelemente Redundanz bestehen kann (in **Abb. 2** dargestellt durch den Beziehungstyp „entspricht“).

Im Folgenden wird auf die Merkmale fokussiert, die potenziell einen Einfluss auf die Architekturauswahl haben.

Antwortzeit und *Zugreifbarkeit*: Antwortzeit und Zugreifbarkeit sind die massgeblichen Bestimmungsgrössen der Pünktlichkeit. Sobald eine der Datenquellen eines Informationsobjekts autonomes Verhalten aufweist, verlängert sich die Antwortzeit bis hin zur Nicht-Zugreifbarkeit. Eine Kompensation dieses Effekts ist (bei unterstellter Nicht-Modifizierbarkeit des Autonomieverhaltens) nur durch eine zusätzliche, redundante Speicherung der betreffenden Datenelemente möglich, wie sie beispielsweise in Data-Warehouse-Systemen vorgesehen ist.

Aktualität: Nicht redundante Datenelemente geben die erreichbare Aktualität der Informationselemente vor. Bei redundanten Datenelementen stellt sich hingegen zu-

nächst die Frage nach ihrer Konsistenz, d.h. ihrer Widerspruchsfreiheit. Bei gegebener Konsistenz kann ein beliebiges der Datenelemente genutzt werden. Bei inkonsistenten redundanten Datenelementen stellt sich hingegen die Frage, welches den aktuellsten Wert repräsentiert; diese Frage lässt sich allerdings nicht mit Sicherheit beantworten. Eine Möglichkeit ist, das Datenelement zu verwenden, das den spätesten Aktualisierungszeitpunkt aufweist.

Vollständigkeit: Die Menge der verfügbaren Datenelemente ist durch den Datenbestand innerhalb der Anwendungsarchitektur typischerweise begrenzt. In der Konsequenz sind einerseits „alte“ Datenelemente und andererseits Datenelemente aus organisationsexternen Quellen nicht in der Anwendungsarchitektur verfügbar. Wenn also entsprechende Informationselemente nachgefragt werden, muss die Datenintegrationsarchitektur einen eigenen Datenspeicher beinhalten, der „alte“ Datenelemente archiviert und Datenelemente aus externen Datenquellen verfügbar hält.

Verwendungsform: Sofern Informationsobjekte nur lesend verwendet werden sollen, ergeben sich kaum einschränkende Anforderungen an eine Datenintegrationsarchitektur. Ein anderes Bild bietet sich, wenn schreibende Zugriffe auf Informationsobjekten vorzusehen sind. Redundanzen zwischen Datenelementen aus unterschiedlichen Datenquellen bedingen bei der erforderlichen Schemaintegration nicht kompensierbare Einschränkungen hinsichtlich der zulässigen Operationen. Ein Beispiel sind bestimmte Formen von Attributwertkonflikten (vgl. [7], [9]), die schreibende Transaktionen unmöglich machen. Ferner ist zu berücksichtigen, dass schreibende Zugriffe bei Redundanz und autonomen Teilsystemen zusätzliche Schwierigkeiten bedingen.

5 Ausblick

Die in diesem Beitrag vorgestellten Merkmale von Informationsobjekttypen und -elementtypen stellen den Ausgangspunkt einer Bewertung von Datenintegrationsarchitekturen dar. Weitere Forschungsarbeit ist erforderlich, um zum einen die einschränkenden Eigenschaften einer Anwendungsarchitektur zu identifizieren; erste Ergebnisse mit Blick auf die Anwendungs- und Datenverwaltungssysteme sowie die Kommunikationsverbindungen (beispielsweise Autonomieverhalten) finden sich in [5]. Zum anderen ist zu untersuchen, welche ökonomischen Konsequenzen einzelne Architekturkonzepte mit sich bringen.

Literatur

1. Alpar, P., Grob, H.L., Weimann, P., Winter, R.: Anwendungsorientierte Wirtschaftsinformatik. 2. Auflage. Vieweg, Braunschweig Wiesbaden (2000).
2. Bleicher, K.: Das Konzept Integriertes Management. Campus, Frankfurt New York (1991).
3. Horváth, P.: Controlling. 6. Auflage. Vahlen, München (1996).
4. IBM Corporation: Business Systems Planning - Information Systems Planning Guide. 4. Auflage. IBM-Form GE20-0527-4, IBM Corp., Atlanta (1984).
5. Jung, R.: Ableitung und Bewertung von Integrationsarchitekturtypen auf Basis fachkonzeptueller Anforderungen, Habilitationsschrift. Universität St.Gallen (2004).
6. Küpper, H.-U.: Controlling. 3. Auflage. Schäffer Poeschel, Stuttgart (2001).
7. Larson, J.A., Navathe, S.B., Elmasri, R.: A Theory of Attribute Equivalence in Databases with Application to Schema Integration. IEEE Transactions on Software Engineering 15 (1989) 4, 449-463.
8. Lehner, W.: Datenbanktechnologie für Data-Warehouse-Systeme. dpunkt, Heidelberg (2003).
9. Lim, E.-P., Srivastava, J., Shekhar, S.: An Evidential Reasoning Approach to Attribute Value Conflict Resolution in Database Integration. IEEE Transactions on Knowledge and Data Engineering 8 (1996) 5, 707-723.
10. Rockart, J.F.: Chief Executives Define Their Own Data Needs. Harvard Business Review 57 (1979) 2, 81-92.
11. Strauch, B.: Entwicklung einer Methode für die Informationsbedarfsanalyse im Data Warehousing, Dissertationsschrift. Difo-Druck, Bamberg (2002).
12 Strong. D.M., Lee, Y.W., Wang, R.Y.: Data Quality in Context. Communications of the ACM 49 (1997) 5, 103-110.
13. Walpoth, G.: Computergestützte Informationsbedarfsanalyse – Strategische Planung und Durchführung von Informatikprojekten. Physica, Heidelberg (1992).
14. Wand, Y., Wang, R.Y.: Anchoring Data Quality Dimensions in Ontological Foundations. Communications of the ACM 39 (1996) 11, 86-95.
15. Zeh, T.: Data Warehousing als Organisationskonzept des Datenmanagements – Eine kritische Betrachtung der Data-Warehouse-Definition von Inmon. Informatik – Forschung und Entwicklung 18 (2003), 32-38.

Anforderungen an eine Architektur zur Integrationsunterstützung in dynamischen Kooperationen aus Sicht der Bauwirtschaft

Thomas Theling, Peter Loos

Johannes Gutenberg-Universität Mainz,
Lehrstuhl für Wirtschaftsinformatik und BWL,
55099 Mainz, Germany
{Theling, Loos}@isym.bwl.uni-mainz.de
http://www.isym.bwl.uni-mainz.de

Abstract. Dynamische Entwicklungen in Unternehmensnetzwerken und deren Umwelt erfordern eine flexible IT-Architektur, die es ermöglicht, auf variable Bedingungen zu reagieren. Dieser Beitrag diskutiert Anforderungen an eine integrative Architektur, die das Management, die Modellierung von Geschäftsprozessen und Kooperationsstrukturen sowie die Prozessausführung betreffen. Als Anwendungsdomäne dient die Baubranche, die mit projektartigen Netzwerken und immanenten Änderungen eine hohe Dynamik aufweist. Die Anforderungen wurden zum einen aus einer theoretischen Betrachtung des Dynamik-Begriffs und dessen Auswirkungen auf Kooperationen erschlossen. Zum anderen wurde eine Befragung von Experten aus Wissenschaft und Praxis zur Erhebung domänenspezifischer Anforderungen durchgeführt.

1 Motivation und Zielsetzung

Die Bildung von Unternehmensnetzwerken ist eine Reaktion auf die zunehmende Globalisierung der Wirtschaft. In der Literatur haben sich einige Theorien zur Kooperationsbildung etabliert, wie bspw. die Transaktionskostentheorie, der market-based view, der resource-based view oder die Spieltheorie. Projektartige Netzwerke sowie sich verändernde Marktbedingungen führen zu einer erhöhten Dynamik der Geschäftsprozesse [1]. Daraus ergeben sich besondere Herausforderungen an die Gestaltung der Informationssystemarchitektur, die zur Integration der am Wertschöpfungsprozess beteiligten Partner herangezogen wird.

Dieser Beitrag diskutiert Anforderungen an eine Architektur zur Integrationsunterstützung in dynamischen Kooperationen und zeigt exemplarische Lösungsansätze für einzelne Probleme. Als Anwendungsdomäne dient die Baubranche, die mit ihren projektartigen Unternehmensnetzwerken, wechselnden Beteiligten und immanenten Planungsänderungen den dynamischen Aspekt von Unternehmensnetzwerken geeignet widerspiegelt. Hierbei werden mehrere Anforderungsebenen betrachtet, die in einem integrativen Gesamtkonzept vereint werden. Kapitel 2 diskutiert zunächst allgemeine Anforderungen an kooperative Architekturen. Kapitel 3 untersucht den Begriff der

Dynamik und zeigt deren Auswirkungen auf die IT. Kapitel 4 stellt Ergebnisse einer Expertenbefragung aus der Domäne der Bauwirtschaft dar. Kapitel 5 zeigt ausgewählte Lösungsansätze. Schließlich fasst Kapitel 6 die Ergebnisse zusammen und gibt einen Ausblick auf weitere Schritte.

2 Allgemeine Anforderungen an Integrationsarchitekturen

In der Literatur existiert eine Vielzahl von Ansätzen, die Architekturen für Unternehmensnetzwerke und Geschäftsprozessmanagement diskutieren. Häufig genannte Anforderungen sind im Bereich der Sicherheit und des Vertrauens anzusiedeln [2-5], wie bspw. eine sichere Transaktionsabwicklung oder die Abbildung von Rollen. Diskutiert wird auch die Flexibilität, die eine einfache Applikations- und DV-Integration, eine Selbstorganisation der Architektur sowie die Skalierbarkeit der Systeme umfasst [3, 6-8]. Die Wiederverwendbarkeit der benutzten Komponenten [6, 9] und die Integration der verwendeten Modellierungssprachen unter Schaffung eines einheitlichen Begriffsverständnisses [3, 10] sind weitere Erfordernisse an die Architektur. Whitescarver et al. [5] erheben weitergehende Anforderungen aus verschiedenen Frameworks für sozio-psychologische Kollaboration, Organisation und Kommunikation, User Interfaces, Netzwerkinfrastrukturen und Metatools. Bemängelter Punkt in klassischen EAI-Lösungen ist das Fehlen geeigneter Monitoring-Instrumente, die integrationsweite Analysen für Entscheidungsträger zur Verfügung stellen [7, 11].

Ein möglicher Ansatz zur Erfüllung der genannten Anforderungen ist die Integration durch Portale mit einheitlichen Benutzeroberflächen, indem sie Werkzeuge zur Team- und Projektarbeit oder zur kollaborativen Prozessabarbeitung anbieten. Portale sollen sowohl unstrukturierte Informationen wie Dokumente oder Grafiken als auch strukturierte Informationen wie Transaktionsdaten, Analysen und Auswertungen verfügbar machen [12]. Einige Ansätze zur Verwendung von Portalen wurden jedoch aus vielerlei Gründen in der betrachteten Domäne nicht akzeptiert.

3 Dynamik und ihre Auswirkungen auf die IT

Dynamik bezeichnet in der Systemtheorie, deren Ansatz im Weiteren verfolgt wird, die Bewegung in einem System bzw. den Anpassungsprozess zur Erhaltung eines Systemgleichgewichts [13]. Die Eigendynamik beschreibt Veränderungen, die von endogenen Faktoren verursacht werden. Dies sind insbesondere systeminterne Ziele, Bedürfnisse und Zwänge, die bspw. durch veränderte Macht- und Einflussfaktoren oder Strategieänderungen hervorgerufen werden.

Die Umfelddynamik beschreibt Veränderungen, die von exogenen Faktoren verursacht werden. Diese Umweltfaktoren sind vom Netzwerk nicht beeinflussbar, wie bspw. politisch-rechtliche, sozioökonomische oder technologische Rahmenbedingungen, welche die Kooperation sowohl fördern als auch einschränken können. Auf die exogenen Faktoren kann nur durch prozessuale oder strukturelle Änderungen reagiert werden. Erkennbar ist ein Zusammenhang zwischen exogenen und endogenen Fakto-

ren. So führen exogene Faktoren bspw. zu neuen Strukturen im Netzwerk, die sich wiederum auf die Prozesse im Netzwerk auswirken [14]. Insbesondere bei Unternehmensnetzwerken, die strukturellen Veränderungen unterliegen, sind gleichzeitig die Prozessstrukturen zu verändern. Diese Strukturveränderungen können ihrerseits durch Prozessveränderungen hervorgerufen werden, so dass eine Interdependenz von Prozess und Struktur erkennbar ist [15].

Der hier dargestellte Zusammenhang zwischen endogenen und exogenen Faktoren sowie Strukturen und Prozessen wird in Abbildung 1 zusammengeführt.

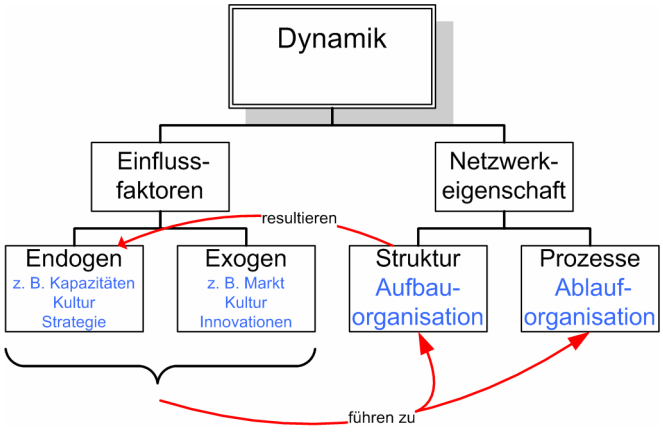

Abbildung 1. Der Dynamikbegriff in Netzwerken

Wesentliche Anforderung, die sich aus der Dynamik ergibt, ist die Flexibilität. Der optimale Grad der Flexibilität ist insbesondere von der Dynamik der exogenen Faktoren abhängig. Hieraus resultieren bspw. die Festlegung und Abbildung von Früherkennungsindikatoren für die Rekonfiguration oder Auflösung des Unternehmensnetzwerks, die Vorbereitung der Aufnahmeentscheidung eines neuen Partners oder die Koordination der Planung von Kooperationsentwicklungsstrategien [16, 17]. Weiterhin ist zu berücksichtigen, dass auf Grund der sich ändernden Prozesse und Strukturen eine Adaption von Modellen und eine Ad-hoc-Abarbeitung von Prozessen gewährleistet werden muss. Dies impliziert, dass aus technologischer Sicht verschiedene operative Systeme flexibel zu integrieren sind.

4 Anforderungen aus der Anwendungsdomäne

Im Rahmen einer Befragung von Experten aus Wissenschaft und Bauwirtschaft wurden Aspekte ermittelt, die bei der Integration vorhandener Konzepte und der Verbesserung abzubildender Geschäftsprozesse in der Bauwirtschaft zu berücksichtigen sind.

Besonders häufig werden Anforderungen im Bereich der Integration genannt. Zum einen werden unterschiedliche Formate für technische Zeichnungen und Pläne verwendet, die zeitnah und aktuell den Unternehmen zur Verfügung stehen müssen. Zum

anderen existieren in der Baubranche Merkmalsdaten sowie Leistungs- und Produktkataloge, die zur Modellierung und expliziten Beschreibung von Leistungen herangezogen werden.

Merkmalsdaten sind Metadaten für Leistungs- und Produktkataloge. Sie strukturieren und definieren Kataloginhalte auf Basis von Merkmalen und Merkmalsausprägungen. *Leistungskataloge* spezifizieren alle Leistungen, die in der Baubranche erbracht werden können (Output). Sie dienen als Referenzdaten für die Leistungsbeschreibung eines Bauprojekts und strukturieren materielle Sachleistungen sowie immaterielle Dienstleistungen (etwa 100 Millionen Leistungen). Sie sind standardisiert (DIN 18299 ff.) und gelten im Rahmen der Vergabe- und Vertragsordnung für Bauleistungen (VOB) als verbindliche Richtlinie (VOB, Teil A, § 9 Nr. 3(4)). Demzufolge müssen sie in operative Systeme zur Planung, Kalkulation oder Beschaffung integrierbar sowie netzwerkweit für jedes Unternehmen verfügbar sein. *Produktkataloge* beschreiben herstellerneutrale Referenzprodukte mittels konkreter Produkteigenschaften. Hierbei handelt es sich um Materialien, die zur Erstellung von Bauwerken verwendet werden (Input). Sie müssen insbesondere während der Kalkulation sowohl für das Gesamtprojekt als auch für einzelne Netzwerkpartner zur Verfügung stehen. Zur Zeit existiert noch kein standardisierter Produktkatalog für die Baubranche, jedoch gibt es erste Bestrebungen in diese Richtung, wie bspw. das Projekt *bau:class* zur Erarbeitung einer Klassifikation für Baustoffe und Bauprodukte [18].

Leistungskataloge und Produktkataloge dienen als Referenzdaten für projektspezifisch zu erstellende Leistungsverzeichnisse und Vergabeeinheiten. Ein *Leistungsverzeichnis* listet die konkreten Teilleistungen und Ausführungsbeschreibungen eines Bauprojekts auf. Es ist Grundlage für die Kommunikation und Basis für die Ausschreibung, Vergabe und Abrechnung (AVA) im Rahmen eines Bauprojektes. Das Leistungsverzeichnis wird i. d. R. zentral von einem Planer erstellt und liefert eine Gesamtübersicht über die zu erbringenden Leistungen. Die *Vergabeeinheiten* sind ähnlich strukturiert, beschreiben aber lediglich die zu erbringenden Leistungen eines einzelnen Unternehmens. Dieses Verzeichnis muss sowohl dem Planer als auch dem Unternehmen selbst in geeigneter Weise zugänglich gemacht werden. Für eine medienbruchfreie Nutzung dieser Daten muss ein standardisierter Datenaustausch der Leistungs- und Produktdaten ermöglicht werden.

Auf Basis der Leistungsbeschreibung muss ein netzwerkweites *Prozessmanagement* etabliert werden. Die Leistungsbeschreibung kann als Grundlage für die Modellierung von Geschäftsprozessen dienen. Einzelne Prozesse werden durch die Vergabeeinheiten eindeutig einem Unternehmen innerhalb des Netzwerks zugeordnet. Es muss gewährleistet sein, dass jedes einzelne Unternehmen Informationen darüber erhält, von welchem Unternehmen die eigene Leistungserbringung abhängt bzw. wer im Gesamtprozess direkt abhängig von der eigenen Leistung ist (Vorgänger/Nachfolger). Die Architektur muss demzufolge die unternehmensübergreifende Modellierung von Geschäftsprozessen unterstützen. Zu beachten ist, dass *unterschiedliche Modellierungsmethoden* sowohl bei den Netzwerkpartnern als auch auf verschiedenen Ebenen verwendet werden. Es sind verschiedene Perspektiven und Abstraktionsstufen zu berücksichtigen: Auf Management-Ebene werden fachkonzeptionelle, stark abstrahierte und semiformale Modelle verwendet, während zu Simulationszwecken und zur Fehlererkennung stärker detaillierte und formalisierte Modelle Anwendung finden.

Hierzu sind geeignete Konverter zu definieren, die eine Transformation semiformaler Modelle in formale Modelle ermöglichen.

Weiterhin muss gewährleistet sein, dass aus Sicht des Unternehmensnetzwerks das Wissen über den Zusammenhang von Leistungen und Prozessen sowie über abstrakte Prozesse vorhanden ist (globales Wissen). Detaillierte Informationen über die Prozesse stellen teilweise Geschäftsgeheimnisse der individuellen Unternehmen dar und sind dagegen nur lokal vorzuhalten (lokales Wissen) [19]. Für die Modelle müssen geeignete Speicher- und Zugriffsmechanismen zur Verfügung stehen, die gleichzeitig ein Versionsmanagement der Modelle unterstützen. Zur Schaffung eines kooperationsweiten Begriffsverständnisses sind einheitliche Konventionen zu spezifizieren.

Aus Sicht des Managements kooperativer Geschäftsprozesse müssen sowohl die Vorplanung von Prozessen zur Build-Time eines Bauprojekts als auch die ständige Kontrolle von laufenden Prozessen zur Run-Time unterstützt werden. Dies betrifft die Analyse, Simulation und Optimierung sowie die Berücksichtigung von Ad-hoc-Prozessen. Für laufende Prozesse müssen Kennzahlen zugänglich sein, um ein kooperationsweites Monitoring zu ermöglichen. Hierzu ist ein standardisierter Datenaustausch zwischen den Monitoring-Tools und den operativen Systemen zu etablieren.

Eine weitere Anforderung ergibt sich aus der Notwendigkeit, mobil und vor Ort auf Informationen zugreifen zu können und diese zu verändern, wie bspw. bei der Mängelerfassung. Besondere Anforderungen sind hierbei die flexible Einbindung neuer Funktionalitäten und Dienste in die Architektur sowie die einfache Einbindung neuer Netzwerkpartner, was typisches Merkmal für die Domäne ist.

Eine Konsolidierung der hergeleiteten Aspekte führt zu folgenden Anforderungen an die Architektur:

1. Modellierung und grafische Darstellung der unternehmensübergreifenden und unternehmensinternen Geschäftsprozesse
2. Analyse, Optimierung und Simulation der Geschäftsprozesse
3. Implementierung der Prozesse in die Systemlandschaft der Unternehmen unter Verwendung bestehender und neuer Anwendungen
4. Ausführung der Prozesse bzw. Unterstützung der Ausführung nicht automatisierbarer Prozesse, sowohl unternehmensintern als auch unternehmensübergreifend
5. Planung und Steuerung der laufenden Geschäftsprozesse

Es resultieren drei Ebenen, auf denen diese Aufgaben anzusiedeln sind: das Management, die Modellierung sowie die Prozessausführung (siehe Abbildung 2).

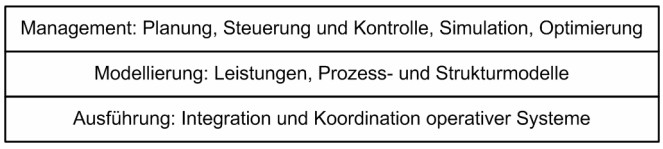

Abbildung 2. Aufgaben-Ebenen der Architektur

5 Lösungsansätze

Als Antwort auf die genannten Bereiche *Management*, *Modellierung* und *Prozessausführung* können bestehende Lösungsansätze wie bspw. Workflow-Management-Systeme oder Geschäftsprozessmanagement-Architekturen identifiziert und in ein gemeinsames Framework integriert werden [20, 21].

Die *Modellierung* nimmt dabei eine zentrale Rolle ein, da auf Basis von Prozess- und Strukturmodellen zum einen durch Monitoring, Simulation und Optimierung Managemententscheidungen herbeigeführt werden können. Zum anderen kann durch geeignete Detaillierung der Prozessmodelle die Ausführung operativer Prozesse zum Teil automatisiert werden. Als Kernkomponente und Datenbasis der Architektur wird daher ein verteiltes Repository vorgeschlagen, das alle anfallenden Leistungs-, Struktur- und Prozessmodelle der Kooperation speichert (globales Wissen) bzw. geeignet referenziert (lokales Wissen). Im Rahmen der Modellierung kommen Referenzmodelle zum Einsatz, die aus dem Repository in die Modellierungswerkzeuge eingelesen werden. Ontologien werden zur einheitlichen semantischen Modellierung genutzt. Zur Speicherung der Prozessmodelle im Repository wird die XML-basierte Business Process Execution Language (BPEL) [22] vorgeschlagen. Sie vereint zahlreiche Charakteristika vorangegangener Standardisierungsbemühungen. Im Repository definierte Rollen beschreiben betriebswirtschaftliche Anforderungen von Personen im Netzwerk.

Zur Unterstützung des *Managements* werden Protokollierungsdaten bzw. phasenübergreifende Daten gesammelt, die orthogonal zu den Modellen stehen. Es wird differenziert zwischen der Build-Time und der Run-Time von Unternehmensnetzwerken. Vorschlag für eine Simulation des Prozess- und Strukturverhaltens im Netzwerk ist die Anwendung der Objekt-Petrinetz-Methode [23]. Sie integriert die modellierten Strukturen und Prozesse, verifiziert die Modelle, bewertet Durchlaufzeiten, Kosten und Ressourcenauslastung und kann zur Optimierung des Unternehmensnetzes a priori beitragen. Instrumente der Run-Time sind bspw. Projektfortschrittskontrollen, die in erster Linie das Prozessmonitoring, die Zeit- und Kapazitätssteuerung oder die Performance-Messung umfassen [24]. Die Auswertungsergebnisse können zum einen zu einer Ad-hoc-Anpassung von Prozessen führen, zum anderen dienen sie für nachfolgende Projektnetzwerke als Wissensbasis für die Modellierung und Ausgestaltung des Netzwerkes.

Zur *Prozesssteuerung* und -integration werden Workflow- und EAI-Funktionalitäten herangezogen. Operative Anwendungen interagieren über dezentrale Applikationsadapter untereinander, ohne eine zentrale Koordinationsinstanz implementieren zu müssen. Als technische Realisierung der Adapter eignet sich die aktuell diskutierte Web Service-Technologie [25]. Die im Repository in BPEL vorliegenden Prozessmodelle unterstützen die Definition und Ausführung von Web Service-basierten Workflows, sofern sie in einem ausreichend detaillierten Abstraktionsgrad gespeichert sind.

6 Zusammenfassung und Ausblick

Dieser Beitrag beschrieb Anforderungen an eine Architektur zur Integrationsunterstützung aus Sicht der Bauwirtschaft. Es wurden zunächst allgemeine Aspekte, Aspekte auf Grund der Dynamik und domänenspezifische Aspekte diskutiert und in einem Ordnungsrahmen den drei Ebenen *Management*, *Modellierung* und *Prozessausführung* zugeordnet. Abschließend wurden exemplarische Ansätze zur Lösung ausgewählter Probleme vorgestellt.

Im Rahmen des vom BMBF geförderten Forschungsprojekts ArKoS – Architektur kollaborativer Szenarien – wurde ein Fachkonzept für die Architektur entworfen, das die Anforderungen der dargestellten Ebenen in einem ganzheitlichen Konzept integriert. Hierauf basierende weiterführende Forschungsfragen verlassen die fachkonzeptionelle Ebene und beschäftigen sich mit der Implementierung und Umsetzung ausgewählter Aspekte der Architektur. Diese beschäftigen sich zum einen mit der Konvertierung semiformaler Prozessmodelle in Objekt-Petrinetze zur Simulation der Modelle. Ein weiterer Schwerpunkt behandelt die Integration von Standardsoftware und die konkrete Ausgestaltungsmöglichkeit der erwähnten Adapter. Aktuell werden Architekturerweiterungen entwickelt, prototypisch umgesetzt und in Showcases getestet. Die gesammelten Erkenntnisse werden sowohl den Aufbau zukünftiger Softwaresysteme als auch Vorgehensweisen bei der Systemintegration beeinflussen. So werden um Geschäftslogik angereicherte aber dennoch offene Schnittstellen (etwa erweiterte Web-Services) den Softwareprodukten hinzugefügt.

Literatur

1. Martin, W.: Die Beständigkeit des Wandels. IT Management (2004) 6, 24-54
2. Megaache, S., Karran, T., Ribeiro Justo, G. R.: A Role-based Security Architecture for Business Intelligence. In: Proceedings der Technology of Object-Oriented Languages and Systems (TOOLS-34"OO) (2000) 295-306
3. Gronau, N.: Kollaborative Engineering Communities - Architektur und Integrationsansätze. In: Loos, P. and Gronau, N. (Hrsg.): E-Business - Integration industrieller ERP-Architekturen. Cuvillier, Göttingen (2002) 1-15
4. De Santis, L., Scannapieco, M., Catarci, T.: Trusting Data Quality in Cooperative Information Systems. In: Proceedings der On The Move to Meaningful Internet Systems 2003: OTM Confederated International Conferences, CoopIS, DOA, and ODBASE (2003) 354-369
5. Whitescarver, J., Bhamidipait, R., Roberts, M. J.: Toward An Architecture for Internet-based 'Evolutionary' Collaboration. In: Proceedings der Third Americas Conference on Information Systems (AMCIS) (1997)
6. Patankar, A.: Web Service Enabled Architecture for Interorganizational Business Process Management. In: Proceedings der Ninth Americas Conference on Information Systems (AMCIS) (2003) 1950-1959
7. Hoffmann, O.: Web-Services in serviceorientierten IT-Architekturkonzepten. HMD - Praxis der Wirtschaftsinformatik (2003) 234, 27-33
8. Noack, J., Mehmanesh, H., Mehmaneche, H., Zendler, A.: Architekturen für Network Computing. Wirtschaftsinformatik 42 (2000) 1, 5-14

9. Blake, M. B.: A Development Approach for Workflow-Based E-Commerce using Reusable Distributed Components. In: Proceedings der Sixth Americas Conference on Information Systems (AMCIS) (2000) 568-573
10. Frank, U.: Eine Architektur zur Spezifikation von Sprachen und Werkzeugen für die Unternehmensmodellierung. In: Proceedings der MobIS-Fachtagung 1999, Modellierung betrieblicher Informationssysteme (1999) 154-169
11. Jeng, J.-J., Schiefer, J., Chang, H.: An Agent-based Architecture for Analyzing Business Processes of Real-Time Enterprises. In: Proceedings der Seventh IEEE International Enterprise Distributed Object Computing Conference (EDOC'03) (2003) 86-97
12. Buck-Emden, R., Böder, J.: Integrierte Geschäftsanwendungen - Service-orientierte Architektur als zukunftsweisendes Modell. Informatik Spektrum 26 (2003) 5, 364-367
13. Ulrich, H.: Die Unternehmung als produktives soziales System. 2. Aufl. Paul Haupt, Bern, Stuttgart (1970)
14. Schwerk, A.: Dynamik von Unternehmenskooperationen. Duncker und Humblot, Berlin (2000)
15. Hippe, A.: Interdependenzen von Strategie und Controlling in Unternehmensnetzwerken: Ansätze für die Steuerung einer auf Kooperation basierenden Organisationsform. Deutscher Universitätsverlag, Wiesbaden (1997)
16. Hess, T.: Netzwerkcontrolling: Instrumente und ihre Werkzeugunterstützung. Deutscher Universitätsverlag, Wiesbaden (2002)
17. Steinle, C., Kraege, R.: Kooperationscontrolling: Eine zukunftsorientierte und phasenbezogene Konzeption der Aufgaben und Instrumente des Controlling strategischer Kooperationen. In: Steinle, C., Eggers, B., and Lawa, D. (Hrsg.): Zukunftsgerichtetes Controlling : Unterstützungs- und Steuerungssystem für das Management. 2. Aufl. Gabler, Wiesbaden (1998) 407-428
18. DIN.bauportal: Memorandum of Understanding (MoU) zur Erarbeitung einer Klassifikation für Baustoffe und Bauprodukte auf der Grundlage des DIN Merkmallexikon (2005). Online: http://www.din-bauportal.de/index.php?mid=60
19. Scheer, A.-W., Adam, O., Hofer, A., Zangl, F.: Nach Cost Cutting - Aufbruch durch Innovation. IM Fachzeitschrift für Information Management & Consulting 18 (2003) Sonderausgabe, 6-13
20. van der Aalst, W. M. P.: Making Work Flow: On the Application of Petri Nets to Business Process Management. In: Proceedings der Applications and Theory of Petri Nets 2002: Proc. of 23rd International Conference, ICATPN 2002 (2002) 1-22
21. Scheer, A.-W.: ARIS - vom Geschäftsprozess zum Anwendungssystem. 4. Aufl. Springer-Verlag, Berlin et al. (2002)
22. Andrews, T., Curbera, F., Dholakia, H., Goland, Y., Klein, J., Leymann, F., Liu, K., Roller, D., Smith, D., Thatte, S., Trickovic, I., Weerawarana, S.: Business Process Execution Language for Web Services 1.1 (2003). Online: ftp://www6.software.ibm.com/software/developer/library/ws-bpel.pdf
23. Valk, R.: Object Petri Nets - Using the Nets-within-Nets Paradigm. In: Rozenberg, G. (Hrsg.): Lectures on Concurrency and Petri Nets: Advances in Petri Nets. Lecture Notes in Computer Science, Vol. 3098. Springer-Verlag, Berlin et al. (2004) 819-848
24. Neumann, S., Probst, C., Wernsmann, C.: Kontinuierliches Prozessmanagement. In: Becker, J., Kugeler, M., and Rosemann, M. (Hrsg.): Prozessmanagement - Ein Leitfaden zur prozessorientierten Organisationsgestaltung. Springer-Verlag, Berlin et al. (2003) 309-335
25. Alonso, G., Casati, F., Kuno, H., Machiraju, V.: Web Services - Concepts, Architectures and Applications. Springer-Verlag, Berlin et al. (2004)

High Availability in a J2EE Enterprise Application Environment

Udo Pletat

IBM Deutschland Entwicklung GmbH
Schönaicher Str. 220
71032 Böblingen, Germany
pletat@de.ibm.com

Abstract

Recent developments of middleware products enabling J2EE enterprise architectures include high availability features in, e.g., J2EE application servers and database systems. This is a step towards making the cornerstones of a J2EE enterprise infrastructure more autonomous, in the sense that their high availability becomes less dependent on dedicated system management software ensuring application and system availability. The article reports about a project exploring recent high-availability features of respective IBM products in an industry application context.

1 Introduction

The trend towards 24x7 operation of enterprise IT infrastructures increases the need to make this IT infrastructure highly available. Besides organizational guidelines how to operate an IT shop continuously, there are many approaches to automate system failovers not only in case of unexpected failures, but also for regular system maintenance or upgrades, see e.g., [1]. The term business resilience - i.e., continuous availability of the IT infrastructure without business relevant downtimes - captures the expectation of IT customers and the challenge for IT providers.

Recent developments of middleware products enabling J2EE enterprise architectures include high availability features in, e.g., J2EE servers and database systems, see [5] and [6]. This makes the cornerstones of a J2EE enterprise infrastructure more autonomous, in the sense that their high availability becomes less dependent on dedicated system management software.
The article reports about a technology assessment project exploring and evaluating recent high-availability features of respective IBM products (WebSphere Application Server[1] and DB2[2]) in an industry application context. The project emphasized obtaining reference results with respect to

- simplifications that can be achieved for setting up a high-availability infrastructure,

[1] WebSphere Application Server is a registered trademark of IBM Corporation
[2] DB2 is a registered trademark of IBM Corporation

- comparing system failover times with high availability configurations relying entirely on dedicated system management software, and
- integration of modern J2EE components of an enterprise architecture with approaches that are operational for 10 years or more.

2 Integration architecture for legacy and J2EE applications

The high-availability setup being discussed reflects an enterprise application integration architecture where legacy application systems are connected to more modern J2EE based applications running on a respective J2EE application server, see also [2]. Figure 1 below shows a typical structure of such an integration architecture.

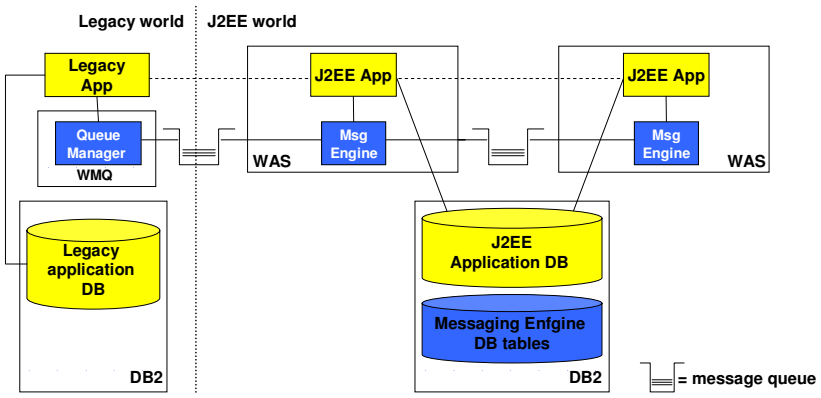

Figure 1: Legacy and J2EE application integration architecture

The fundamental components of such EAI structure are

- a (set of) legacy application(s) attached to a JMS-free message queuing infrastructure, e.g., based on message queuing systems like WebSphere MQ[3];
- the J2EE applications are hosted on respective application servers. Modern servers like WAS 6 include full JMS provider functionality capable to interact with respective non-JMS message queuing systems;
- the J2EE applications need to access databases, e.g., based on DB2. The database system holds the application data and serves as the persistent message store maintaining respective tables of the messaging engine[4].

The key objective of setting up a high-availability infrastructure for the J2EE part of such an enterprise integration architecture is a deployment of the J2EE part of the above enterprise application to a redundant hardware topology, where each J2EE application server is clustered with the cluster typically containing a primary and a standby machine; the same applies to the database server. In order to achieve short

[3] WebSphere MQ is a registered trademark of IBM Corporation

[4] The database view is slightly simplified for the time being, but there is no conceptual difference whether to have a single DB shared by multiple J2EE servers or each J2EE server accessing its 'own' DB or having criss-cross database access. Typically, one may assume that the legacy applications run against their own database.

failover times, a so-called 'hot standby' approach is mandatory; in such a configuration the primary server performs all normal operation while the standby server is ready to take over, but idles until the failover becomes necessary.

3 High-availability scenarios and requirements

Typical high-availability scenarios to be covered are

- network disconnections between the legacy applications and J2EE application cluster or between the J2EE cluster components,
- failure of a J2EE application server, and
- database server failure.

The kernel requirements for a high-availability system setup are usually

- ease of setup and operation and
- short failover times with (close to) no human intervention.

The 'classical approach' to cope with these high-availability requirements is to use shared disks holding the relevant data of the middleware component to be made highly available and two server nodes - primary and standby – accessing the component's data on the shared disk, see also [13]. In case of a failover, high availability management software like Tivoli System Automation (TSA)[5] or High Availability and Cluster Multi-Processing (HACMP)[6] (see [9] and [10]) starts the failed middleware component on the standby machine. This failover process makes sure that all relevant file systems, IP addresses, etc. for that component are switched over to the standby node, see left hand part of Figure 2 below illustrating the shared disk concept for the database failover.

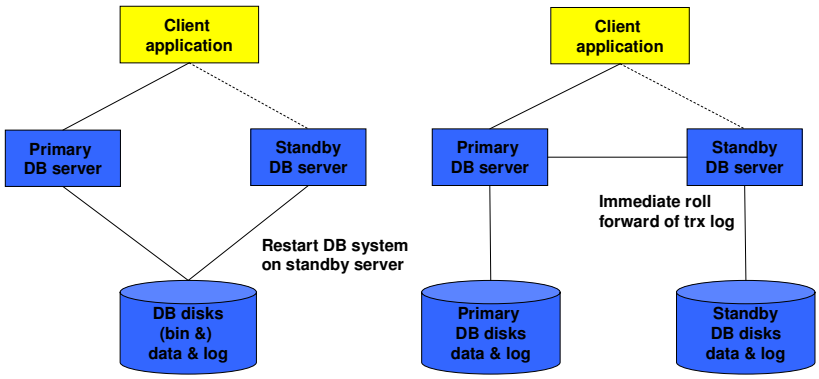

Figure 2: Shared disk plus restart versus transaction log replication

The right hand part of Figure 2 above shows how data replication features in a product like DB2 8.2 can be used for making the database system highly available:

- in-transaction data replication through log shipping from the primary to the standby database server assures that the standby database keeps track with the DB operations executed against the primary database;

[5] Tivoli System Automation is a registered Trademark of IBM Corporation

[6] High Availability and Cluster Multi-Processing is a registered Trademark of IBM Corporation

- automatic client re-route assuring that the database failover is basically transparent to client applications.

The transaction log forwarding approach - also known as 'eager data replication', see [11] or [12] - combines aspects of high-availability and data replication for disaster recovery scenarios. Looking a bit closer at the options for a database high-availability setup, Table 1 summarizes some arguments for/against the two main approaches.

Table 1: Shared disk versus replication approach to database high availability

	Pro	Con
Shared disk	- no performance penalty in normal operation	- dependence on storage area network - database corruption due to failure may require database restore - more complicated system management software configuration
Data replication	- simpler system management software setup - database restores after primary database failure are less likely	- performance penalty in normal operation

For the database side the overall failover time, i.e., the time between the occurrence of a failure and the point in time when the applications resume processing after the component failover is complete, is not so much influenced by the database startup times. However, in a complicated storage area network setup for high volume databases, the disk switching may consume a significant portion of the failover time.

On the WebSphere side, the main argument for the hot standby approach is indeed the shorter failover time because there is no need for restarting the application server and the deployed applications on the standby node. Application server start time plus application loading time can be significant, so the availability of the standby application server with all applications being started there – but remaining passive until a failover occurs – is an essential decision criterion.

High-availability features of a product like WebSphere Application Server Version 6 include

- setup of clusters consisting of redundant application server nodes with failover policies like '1 of N' (meaning that application server runs on 1 out of N available server machines), 'static' (meaning that application server has to run on a dedicated server machine), plus user-definable policies to let also external high-availability management software interact with WebSphere Application Server,
- support of so-called *singleton services* that run only once in a WAS high-availability cluster; typical singletons are the messaging engine and the transaction manager;
- improved failover times compared to using a shared disk approach plus typical high-availability management software.

These new features reduce the need for using dedicated high-availability management software and move J2EE server high-availability away from being 'only' based on workload management features where - in a J2EE server cluster - the workload of a failed server will be taken over by other servers in the cluster.

4 High Availability using recent WebSphere and DB2 technology

In the previous section we have outlined some key features of recent middleware products and now we sketch how these features can be exploited for setting up a highly available J2EE server infrastructure where

- high availability requirements of the J2EE server can be satisfied with the WAS internal high availability features[7]. The key element being the hot-standby application server which can takeover immediately without any need for server re-starts and replaying transactions other than those that were 'in-flight' at the point in time the primary server failed.
- high availability features of the database for the J2EE server(s) are fulfilled by using the high availability and disaster recovery feature of DB2. For database failover, this means that we also arrive at a hot-standby setting where at most the in-flight transactions have to be replayed on the backup database server after the failover has occurred[8].

The next sections provide some details about the respective setup of WAS and DB2.

4.1 WAS System setup

The layout of the so-called WAS cell is illustrated in Figure 3 below. The setup for the study consists of a 5-node cell where the nodes on the left-hand side make up the so-called primary tower while the two nodes on the right-hand side make up the backup tower, see Figure 3 below. The transaction log files for each J2EE server pair (primary and backup) reside on a NFS Version 4 enabled file system, which can be attached either in NAS-style (Network Attached Storage through a respective file server – which in our case also hosts the deployment manager administrating the WAS cell configuration) or SAN-style (Storage Area Network via directly attached disks).

WebSphere controls the failover of the application server itself plus the transaction manager and messaging engine. Using the so-called 'connection proximity' configuration allows to specify that an application can only participate in message traffic, if a messaging engine is hosted by the application server where the application is deployed. This allows for having all applications started both on the primary and the standby application server; the availability of the messaging engine on either of the cluster's application servers defines whether of the application

[7] When WAS fails over a connectivity loss between the WAS Messaging Engine and the WebSphere MQ Queue Manager cannot be recovered automatically by WAS alone, because WAS does not support an 'automatic client-re-route' feature as offered by DB2. Respective TSA scripting failing over the WAS IP address fills this current gap.

[8] Also here there is a little gap in the sense that the triggering the takeover of the backup database fully automatically requires a simple use of high-availability management software like TSA. This automates the decision on whether the primary database server is considered down so that a failover has to be initiated.

deployed on the primary or standby application server may process messages, thus making the applications 'quasi-singletons'.

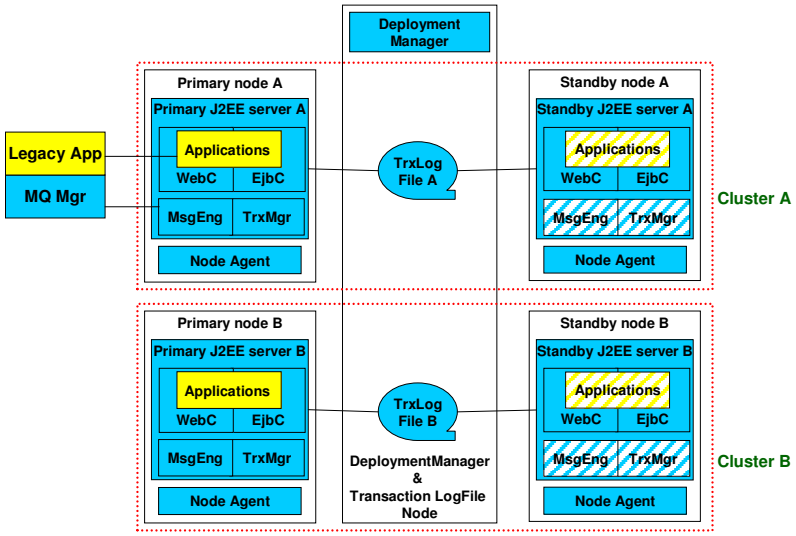

Figure 3: High availability layout of the J2EE server cell

4.2 DB2 System Setup

Exploiting the HADR (high-availability and disaster recovery) feature of DB2 leads to a quite straight-forward setup of the highly available database, as illustrated in the right hand part of Figure 2 above. The database system has to maintain both the J2EE application data and it also serves as the persistent store for the Messaging Engine. In most cases it is a recommended practice to separate the application data from the Messaging Engine data leading to at least two databases for which a primary and standby instance have to be configured for use with HADR. There are three reliability modes for transferring the transaction logs from the primary database node to the standby:

- *'synchronous'* - the original operation on the primary database succeeds when it receives a notification from the standby database that the transaction log entry has been written to the disk on the standby server;
- *'near synchronous'* - the original operation on the primary database succeeds when it receives a notification from the standby database that the transaction log entry has been written to the main memory on the standby server;
- *'asynchronous'* - the original operation on the primary database succeeds when it has transferred the log file entry to its TCP/IP system.

The three modes allow for balancing between performance and reliability requirements as the transfer of the transaction log data from the primary to the standby database server is part of the original operation executed on the primary server.

According to the database replication scheme proposed in **[12],** the HADR approach falls into the category of so-called *'eager replications'* and especially into the category *'primary copy, linear, voting'* (for the synchronization modes 'synchronous' and 'near synchronous') and *'primary copy, linear, non-voting'* (for the synchronization mode 'asynchronous'), respectively.

4.3 Making your applications HA-aware

Of course, applications do not become highly available by 'simply' making the middleware infrastructure to which they are deployed highly available. In the preceding sections we have 'only' described the prerequisites that have to exist anyway in order to prepare your IT infrastructure for making the applications highly available.
As outlined also in [1], the applications must be able absorb failing accesses to the middleware in case, e.g., the database becomes unavailable for a certain period of time. Typical strategies include

- enabling access retries;
- saving consistent intermediate state to disk so that in case of a failover and application restart on the backup node, the new instance of the application knows the state of the old one before that one 'died';
- capturing exceptions caused by the failovers appropriately, e.g., DB2 supports an automatic client re-route where applications will be notified about a database failover through respective exceptions;
- fnally, the transaction structure for processing incoming messages has a high impact on which high-availability provisions have to be taken in the application code. Especially the message processing may vary from using Message Driven Beans together with Container Managed Transaction (simple) to using plain JMS and the Java Transaction API (complex), see also [4] and [5].

The table below gives a quick overview on the various transaction management alternatives that are of interest when the message processing shall take place under transactional control in a J2EE enterprise application, see, e.g., [3] and [5].

Table 2: Transaction control alternatives

	Transaction control mechanism	**Message consumption**	**Failover behavior**
JMS	Java Transaction API	inside transaction	- transaction rolled back - message back to queue after rollback
	Java Transaction API	outside transaction	- transaction rolled back - message no longer on queue after rollback
MDB	Bean managed transactions with JTA transaction demarcation	outside transaction	- transaction rolled back - message no longer on queue after rollback
	Container managed transactions with transaction mode 'required' or 'notSupported'	inside transaction	- transaction rolled back - message back to queue after rollback

To exploit the transaction recovery services of the J2EE server after the failover is accomplished, it is a good practice to consume messages within the transaction scope – no matter whether using JMS or MDB. Whether to use JMS or MDB depends on the degree of flexibility of the transaction structure for processing messages required by the J2EE applications.

5 Results and experiences

The results of this advanced study exploring the high availability features of recent middleware products being the key elements of typical J2EE base enterprise IT architectures are both promising and creating new expectations.
On the positive side we find simplified setups of a highly-available J2EE infrastructure, including database access. In addition the so-called 'hot-standby' servers are becoming reality with shortened failover times, especially for J2EE application servers.
On the other hand, new expectations are also being created, e.g., by asking for improved failover policies both for WebSphere Application Server and DB2; finally, integration with upcoming trends of on-demand operating environments which improve the dynamic management of middleware infrastructures may lead to the next generation of high-availability approaches.

Acknowledgement
The author would like to thank Michael Haeberlen, Andrew James, and Sven Stueven for their contributions to bring the system described in the article to life.

References

[1] Blueprints for High-Availability, E. Marcus and H. Stern, Wiley Publishing Company, 2003
[2] Patterns of Enterprise Application Architecture, M. Fowler, Addison-Wesley Publishing Company, 2003
[3] Java Message Service, Version 1.1, Sun Microsystems Corporation, 2002
[4] Java Transaction API, Version 1.0.1b, Sun Microsystems Corporation, 2002
[5] Enterprise Java Beans, R. Monson-Haefel, O'Reilly Publishing Company, 2002
[6] WebSphere Application Server Version 6, Information Center, IBM Corporation, 2004
[7] DB2 Version 8.2, Data Recovery and High Availability Guide and Reference, IBM Corporation 2004
[8] IBM WebSphere Version 6: Performance, Scalability and High-Availability, IBM Redbook, IBM Corporation 2005 (draft)
[9] Tivoli Systems Automation for Multi-Platforms, Guide and Reference, Version 1.2, IBM Corporation 2004
[10] High Availability and Cluster Multi-Processing for AIX, Version 5.1, IBM Corporation, 2003
[11] The Dangers of Replication and a Solution, J. N. Gray, P. Helland, D. Sasha and P. O'Neil, Proceedings 1996 ACM SIGMOD International Conference on Management of Data, pp 173-182, Montreal, June 1996
[12] Database Replication Techniques: A Three Dimension Comparison. M. Wiesmann, F. Pedone, A. Schiper, B. Kemme, and G. Alonso, Proceedings 19th IEEE Symposium on Reliable Distributed Systems (SRDS 2000), Nuernberg, Germany, October 2000
[13] IBM WebSphere Version 5.1: Performance, Scalability and High-Availability, IBM Redbook, IBM Corporation 2004