Virtual Decoupling for IT/Business Alignment – Conceptual Foundations, Architecture Design and Implementation Example

IT/business alignment is one of the main topics of information systems research. The main challenge in this respect ist the mutual alignment of business and IT structures. Services, applications, and domains constitute fundamental artifacts and are described for the proposed alignment architecture. Direct equivalents for the elements of the alignment architecture are missing at both business and IT side. They should be of coarser granularity than those artifacts that are linked by them. Existing methods can be integrated into the proposed approach. As an alternative a bottom-up approach is proposed and demonstrated by means of a case study for the building of domains as virtual, aggregated artifacts of the alignment architecture.

DOI 10.1007/s12599-008-0010-7

The Authors

Dr.-Ing. Stephan Aier
Prof. Dr. Robert Winter
Institut für Wirtschaftsinformatik
Universität St. Gallen
Müller-Friedberg-Strasse 8
9000 St. Gallen
Switzerland
{stephan.aier | robert.winter}@unisg.ch

Received: 2008-05-01
Accepted: 2008-09-02
Accepted after two revisions
by Prof. Dr. Buhl.

This article is also available in German in print and via http://www.wirtschaftsinformatik.de: Aier S, Winter R (2008) Virtuelle Entkopplung von fachlichen und IT-Strukturen für das IT/Business Alignment – Grundlagen, Architekturgestaltung und Umsetzung am Beispiel der Domänenbildung. WIRTSCHAFTSINFORMATIK. doi:10.1007/11576-008-0115-0.

1 Introduction

Integration is often regarded as a primary subject for design-oriented information systems research (Rosemann 1999; Vogler 2003). Integration is a complex concept, characterized by a variety of dimensions including a variety of attributes. Common integration dimensions are

- Integration type (merging vs. linking),
- Integration scope (intra-organizational vs. inter-organizational),
- Integration object (data vs. functions vs. processes) or
- Integration orientation (vertical vs. horizontal).

Mertens (2007, pp. 1–10) provides a comprehensive overview of integration dimensions and attributes.

Among the different types of integration, the integration of business requirements with implemented IT functionalities is of particular interest. According to (Luftman 2005) “IT/business alignment” has been the most important topic for CIOs across various industries and company sizes for several years. The integration of business requirements and IT implementation is complex. (Henderson and Venkatraman 1993) propose a matrix where “strategy” and “processes” are differentiated on a vertical axis, while “business view” and “IT view” are differentiated on a horizontal axis. Based on the resulting four quadrants “business strategy”, “IT strategy”, “business processes & infrastructure” and “IT processes & infrastructure”, four different integration perspectives can be differentiated.

Integration can be achieved by merging and by linking (Rosemann 1999, p. 5 f). Different responsibilities, high complexity and limited flexibility of legacy systems often prevent integration by merging. For example, if information systems of different companies support the same inter-organizational business process within a value chain, merging integration becomes impossible due to the distributed responsibilities and different regulatory requirements of the parties involved. If different artifact types such as business requirements and IT functionalities have to be aligned, merging integration is also not possible. Even if merging integration was possible, it would not in every case be desirable in order to avoid the evolution of monolithic and hard-to-maintain information systems.

If integration is not realized by merging, linkage is the other option. In this case mapping mechanisms are of great importance. For example, IT/business alignment mechanisms need to map strategic business artifacts (e. g. products, goals), organizational business artifacts (e. g. processes, organizational structures) and IT implementation artifacts (e. g. data structures, software components).

So far, integration of artifacts using point-to-point relationships seems to be an obvious solution. However, in complex integration environments, it can be expected that each artifact involved is part of multiple point-to-point relationships. In these complex artifact structures, a change to a single entity causes a multitude of necessary subsequent adjustments. An example can be found in Cobol programming. When changing the processing logic of a program, it might become necessary to change of the data structures used (“read”). Furthermore, the structure of the processed file ("write") is changed, which causes modifications in all other programs that need to parse this data. With growing complexity of the structures involved and increasing change frequency of the entire system, point-to-point relationships become more and more inefficient. A solution to this problem is the establishment of indirect m:n relationships by using a "hub". A good example is the Enterprise Application Integration (EAI) concept (Kaib 2002; Linthicum 2000) which establishes an integration infrastructure for m:n linkage of software components. A multitude of point-to-point connections between software systems is replaced with a linkage system to which each software system has one single point-to-point connection ("adapter"). This implies that many direct connections are replaced with few adapters.

However, EAI only links software systems and other hub-and-spoke architectures also focus on linking artifacts of the same kind. IT/business alignment must link different types of artifacts (e. g. process activities must be linked to software components). Therefore, m:n relationships should be established by means of a central mapping infrastructure. From an architectural point of view such a mapping mechanism is placed "between" business and IT structures.

The goals of this article are to

1. demonstrate that IT/business alignment can be supported by linking integration using an m:n mapping mechanism, which will achieve significant improvements compared to point-to-point connections
2. derive the architecture for this mechanism from analogue mechanisms developed in systems theory, computer science and economics
3. propose mapping artifacts to enable linkage of business and IT artifacts using indirect m:n relationships
4. show that besides a top-down approach to design mapping artifacts, bottom-up design is reasonable as well
5. demonstrate the contribution by presenting a case study of a telecommunications supplier using the bottom-up approach for domain identification as mapping artifacts.

The structure of the article follows a process model in design-oriented information systems research (Rossi and Sein 2003):

- *"Identify a need"*: The necessity of an m:n mapping mechanism "between" business architectures and IT architectures is identified in the first section. Section 2 gives a brief overview of the terminology and the architectural understanding used by the authors. The available knowledge of similar mechanisms in systems theory, computer science and economics is discussed in section 3. The extent to which existing mapping mechanisms can be adapted for IT/business alignment is analyzed on the basis of findings in related work.
- *"Build"*: Existing mapping mechanisms for artifacts, such as the three-layer database architecture or market mechanisms for standardized goods, are adapted for the integration of business architectures and IT architectures. The artifact types needed are outlined in section 4. It is shown that most approaches to designing artifacts follow a top-down direction. A bottom-up approach is presented in section 5.
- *"Evaluate"* (Section 6): The bottom-up design of mapping artifacts is demonstrated by presenting the implementation of a prototype for domain clustering. The feasibility of the proposed approach is evaluated in respect of the design goals defined in sections 1 and 2.
- *"Learn & theorize"* (Section 7): The results of the (partial) evaluation as well as implications of the presented approach are discussed. In particular, implications for integration management are considered.

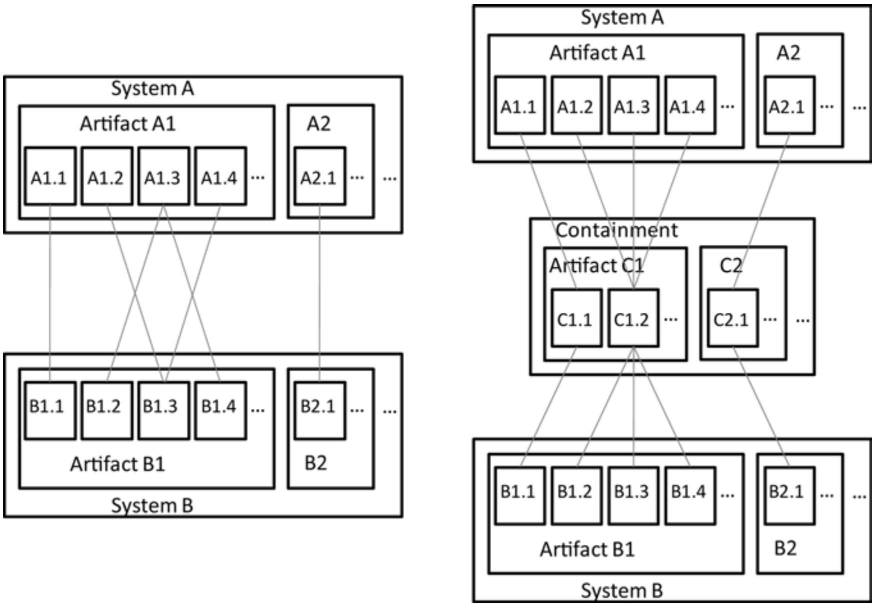


Fig. 1 Generic m:n mapping mechanism in systems theory

2 Terminological foundation

Product and service descriptions, goal definitions, process specifications, organizational structures, etc. are business artifacts and therefore covered by the business architectures. Software components, data structures, etc. are IT artifacts documented by IT architectures. Contrarily to these, the above described mapping mechanism for IT/business alignment is a virtual artifact for the purpose of "translating" between the two architectures. It is neither part of business architecture nor of IT architecture.

The fundamental artifacts used to build mapping mechanisms are called enterprise services. Enterprise services serve to map

- IT functionalities (e. g. functionalities implemented by software components and data structures) on the one hand with
- activities (business structures necessary for process execution) on the other hand

indirectly and with n:m capability. Enterprise services must be of coarser granularity than the linked business functionalities and activities to prevent the flexibility advantage gained (compared to point-to-point connections) being outweighed by the creation of a highly complex additional architectural layer.

In addition to enterprise services as fundamental mapping artifacts, organizations use applications and domains to support the top-down approach and the communication of the mapping architecture. Applications and domains are virtual artifacts as well, since they support IT/business alignment and are neither part of business architecture nor of IT architecture.

The architectural layer representing enterprise services, applications and domains is called the *alignment layer*. The artifacts within the alignment layer are referred to as *alignment artifacts* and form the *alignment architecture*. The alignment layer is introduced to complete the architectural representation of an organization in addition to one or more business layers (representing products, goals, processes, organizational structures, etc) and one or more IT layers (representing software components, data structures, etc.), and therefore implement the integration needs of IT/business alignment by m:n linkage. Due to the virtual nature of artifacts of the alignment layer, special requirements and challenges arise for the design and management of such artifacts. Despite 'only' being represented by virtual artifacts, the alignment architecture is no less important than business-related and IT-related architectures. While a business architecture is designed in order to achieve process effectiveness and process efficiency (Winter 2008) and while IT architectures strive for technical design goals such as reusability, scalability and performance, the alignment architecture has its very own design goals, such as e. g. creation of transparency, (linking) integration, simplification and/or flexibility (Winter and Schelp 2006).

3 Decoupling mapping mechanisms in systems theory, computer science and economics

This section introduces basic principles of mapping mechanisms using the knowledge and vocabulary of general systems theory. Applications of this concept in computer science and economics are subsequently presented.

3.1 Decoupling mapping mechanisms in systems theory

The problem of complexity is the starting point for decoupling in systems theory. The complexity of the environment is always higher than the complexity of the system in focus. Due to limited capacity, the system is not able to establish direct point-to-point relationships to each environmental element. Possible ways to deal with this problem are ignorance (not responding to certain environmental aspects) or bundling and generalization (responding in the same way to different environmental aspects).

Generalization intercepts the direct point-to-point relationships by bundling on the system side or on the environment side. Systems that intercept disturbances of the environment locally without distributing the disturbances to other parts of the system are called ultrastable systems (Ashby 1981, p. 48). The interdependencies of system components can be reduced or intercepted by using multistage systems (Luhmann 2002, p. 169). This interception of dependencies is also referred to as 'loose coupling' (Glassman 1973; Luhmann 2002, p. 171; Weick 1976). The opposite of 'loose coupling' is 'tight coupling' which does not usually occur in natural systems.

The greater the extent to which tight coupling is implemented within a system, the more vulnerable the system is to changes: Disturbances to components can influence the entire system. This problem was especially addressed in the discussion on high technology systems, since 'tight coupling' approaches were often implemented in this context (Luhmann 1991; Perrow 1999). Systems theory proposes the "containment" concept to overcome this problem: A containment can serve as wrapper which again follows the principles of 'loose coupling' (cf. **Fig. 1**) (Luhmann 2002, p. 171).

Fig. 1 illustrates the impact of a containment. If the basic artifacts of the containment are more coarse-grained than the decoupled artifacts, then direct m:n relationships (left side) will be replaced by m:1 and 1:n relationships on the right side.[1] The containment represents the semantic of the original, direct m:n linkage and prevents every change in system A from having to be propagated into system B.

With the containment artifact, systems theory justifies the benefit of a loose coupling mapping on a generic level. Using the containment mechanism enables artifacts of the same type as well as artifacts of different types to be decoupled. Containment artifacts can be implemented by means of real or virtual artifacts. In the following, application examples of this generic linkage mechanism found in computer science and economics are presented. These relate exclusively to cases where different types of artifacts are decoupled using a virtual mapping architecture.

3.2 Decoupling mapping mechanisms in computer science

An obvious example of an m:n capable, virtual decoupling layer as claimed in section 1 and generically described in section 3.1 is the ANSI/X3/SPARC 1975 standard for databases. This guideline, also known as three-layer database architecture, requires that a conceptual layer serves for decoupling purposes 'between' data usage models (external layer) and data implementation models (internal layer). Instead of directly linking implemented data structures (e. g. records, indices) to data usages (e. g. figures in reports), a conceptual data model serves as a virtual linkage layer. This allows an additional view on the database architecture where data storage and data usage can be modified independently (logical and physical independence) (Date 2000). The decoupling benefit of this mapping architecture is sufficiently high to justify the creation and maintenance efforts for the additional layer.

Business requirements and IT systems tend to change autonomously, e. g. due to regulatory changes (business requirements) and release changes for standard software (IT systems). As a consequence, IT/business alignment is a continuous process which can be based on the integration architecture. Similarly, autonomous changes occur for data implementation (e. g. technical performance optimization) and data usage (e. g. query changes). The conceptual data model integrates all possible queries and implementations. As a consequence, the integration architecture for IT/business alignment should also be independent from implementations in order to be able to represent as many requirement variants as possible as well as to ensure decoupling and create the business-focused analogy of physical and logical data independence.

[1] Replacing direct m:n relationships with m:1 and 1:n relationships represents a specific solution. A general solution for coarsening is the replacement of direct m:n relationships with m:x and y:n relationships, where m>x and y<n. The following figures illustrate special solutions for the purpose of simplification; however, the explanations also remain valid for a general solution.

Deutsche Telekom Laboratories

We shape the future

Deutsche Telekom Laboratories is Deutsche Telekom's research and development institute based in Berlin. It is simultaneously a scientific institute organized under private law and associated with the Technische Universität (TU) Berlin. At Deutsche Telekom Laboratories, scientists from across the globe work together with experts from the Group to develop new services and solutions for Deutsche Telekom's customers. Establishing new companies (spin-offs) is another method for making use of research output.

Cooperation with the TU Berlin, other universities and industry partners creates a bridge between business and science in order to turn ideas into marketable innovations as quickly as possible. As part of this, Deutsche Telekom Laboratories focuses on five fields of innovation (5 i):

- **I**ntuitive Usability of services and devices
- **I**ntegrated Service Components
- **I**ntelligent Access
- **I**nfrastructure for IT and telecommunications
- **I**nherent Security

The business and information systems engineering offers useful interdisciplinary approaches for all these areas of innovation. Subject matter includes, for example, modeling, methods and tools for process innovations, agile architectures for information and communication technologies (ICT), technology-oriented management approaches and techno-economic assessments. The aim is to safeguard the economic sustainability of innovations for the Group.

Deutsche Telekom Laboratories is divided into two areas: The Innovation Development Laboratory focuses on market-centric research and development within a timeframe of up to three years. The basic and technology research of the Strategic Research Laboratory has a long-term focus. Common goal: Deutsche Telekom Laboratories is looking to become one of the world's leading research and development institutions in the field of new ICT.

An institute was set up together with Ben-Gurion University in Beer Sheva, Israel, in 2006. Since 2008, Deutsche Telekom Laboratories has also been represented in Darmstadt. Another project office was opened in the Silicon Valley, United States, in January 2009.

Contact:
Deutsche Telekom Laboratories
Ernst-Reuter-Platz 7
10587 Berlin, Germany
E-mail: wi.laboratories@telekom.de
www.laboratories.telekom.com

Deutsche Telekom Laboratories
An-Institut der Technischen Universität Berlin

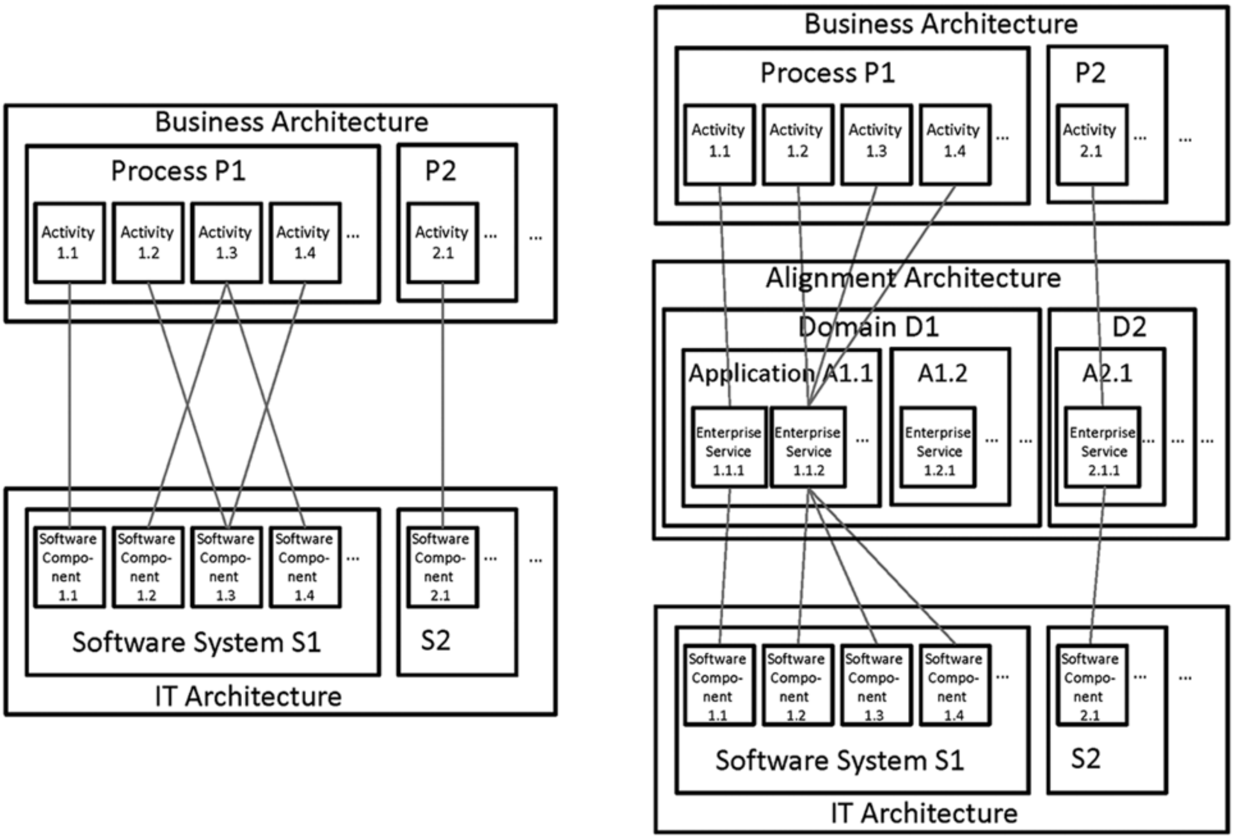


Fig. 2 Alignment architecture for a decoupling mapping of business and IT artifacts

Other IT-related linkage architectures such as e. g. the data warehouse system (Devlin 1997; Inmon 1996) or Enterprise Application Integration (Kaib 2002; Linthicum 2000) are less appropriate examples in terms of the requirements defined in the first section: These mechanisms decouple artifacts of the same type which are not subject to autonomous changes on different architectural layers.

The findings of the ANSI/X3/SPARC example can be summarized as follows:

- An additional architectural layer ‘between’ data usage and data implementation is created.
- This architectural layer primarily serves as a “translator” of the linked architectures and belongs neither to data usage nor to data storage.
- The additional architectural layer decouples artifacts of different types on different architectural layers.
- Direct m:n relationships are replaced by m:1 and 1:n relationships.
- Modifications of artifacts on the decoupled layers do not necessarily need to be propagated.

3.3 Decoupling mapping mechanisms in economics

From the economic perspective, the market is a good example of a mapping mechanism. The market organizes the matching of participants’ needs and maps the supply and demand for goods. Therefore, it is not necessary for each market participant, e. g. with a certain demand, to find an appropriate counterpart (and offer to supply accordingly), to establish direct point-to-point relationships and to negotiate e. g. on price and product definition. Assuming a certain number of market participants on both the supply and the demand side, it is more efficient to use the market as a mapping architecture (cf. Ricardo 1830; Smith 1776). Besides classical market functions such as price definition, market clearing, allocation and efficiency, market functions such as flexible adjustment to changing environmental factors are of particular interest (Fritsch et al. 2005; Herdzina 2005; Kantzenbach 1967; Kerber 2007). On both sides of the market (supply and demand), certain changes can be made without effects for the other side. The market partially balances and compensates for these changes. For example, if individual suppliers enter or exit the market, that does not necessarily have effects on the market as a whole since buyers can usually still satisfy their demand. Vice versa, changes in demand structure do not necessarily have an effect on the suppliers. In the classical sense, markets are locations (marketplaces) for the trade of goods and services. Today, markets are increasingly virtual, which makes decoupling even more effective, since more participants on the supply and demand side operate within the market (Kollmann 2001; Weiber and Kollmann 1998). Of course, this interpretation of market mechanisms is strongly simplified as certain market functions such as price definition and product standardization are not considered.

The similarities between the market example and the computer science example based on the general systems theory concept become more apparent when product standardization is taken into

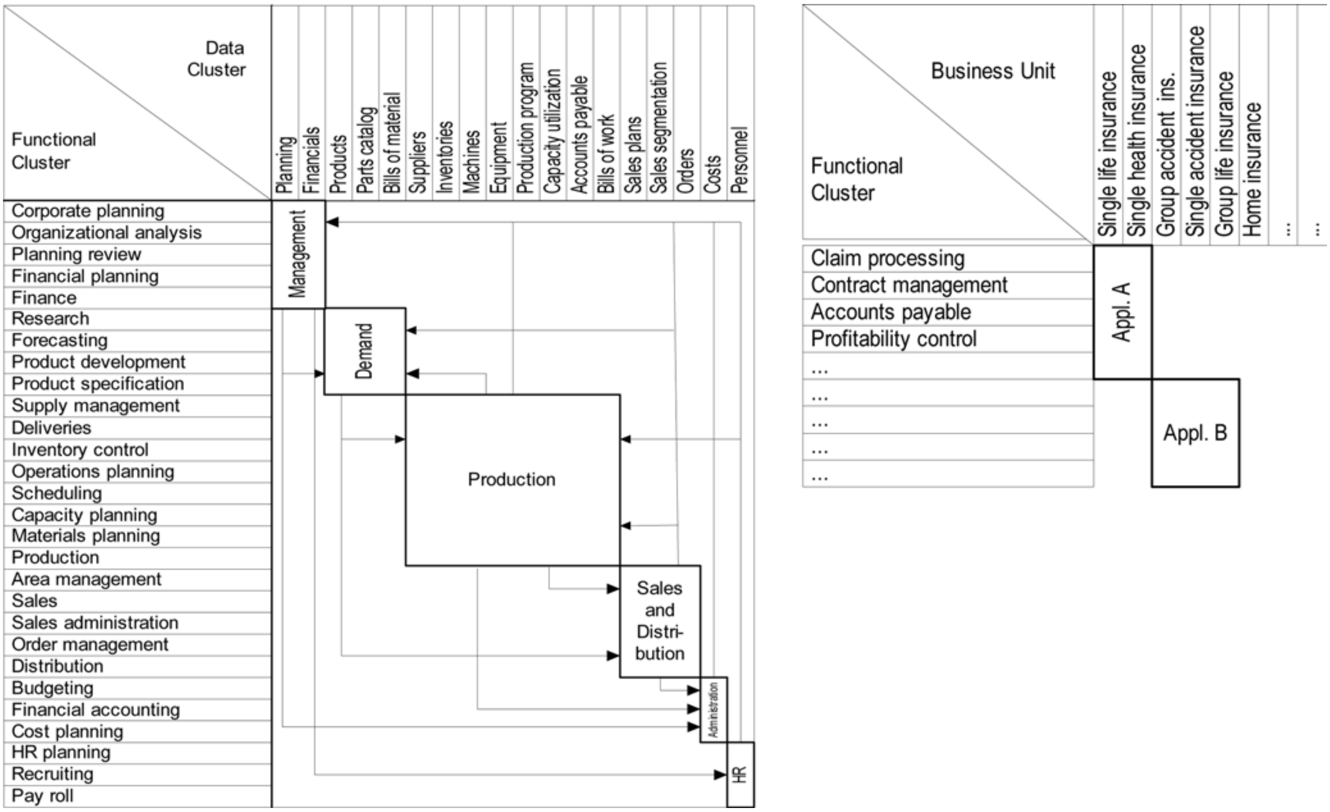


Fig. 3 BSP matrix and STP matrix

consideration. Standards can constitute a mapping architecture for all kinds of artifacts. Many examples of standard-enabled decoupling can be found in the domains of information systems (Buxmann and König 1998; Buxmann et al. 1999; Meffert 1993) or process design in business-to-business environments (Dietrich 2008). Standardization incurs implementation costs and cost to develop standard-compliant artifacts, and promises benefits from the interchangeability of standardized artifacts (Buxmann and König 1998; Shapiro 2001). The more users rely on a standard, the greater is the chance that benefits will exceed implementation costs (Katz and Shapiro 1994). However, when standards remain very generic and less specific, the adaptation efforts are usually higher, and the immediate benefits are usually lower.

The most important characteristics of the market example (for standardized goods) can be summarized as follows:

- An additional architectural layer ‘between’ supply and demand is created.
- This layer may be, but does not necessarily have to be virtual.
- The layer enables decoupling of different types of artifacts.
- Direct m:n relationships are transformed into m:1 and 1:n relationships.
- The artifacts on the decoupled layers can often be modified without effects for the other layers.
- The implementation of the additional layer causes additional efforts.
- The benefits of the decoupling layer increases as the number of linked parties rises (network effects).
- The more specific the artifacts of the decoupling layer, the less benefit is gained from this layer.

4 A Decoupling mapping mechanism for IT/business alignment

Based on the requirements presented in the first section and following the examples outlined above, a mapping mechanism to flexibly support the integration of business and IT is introduced in this section. In a first step, requirements for the architecture and the artifacts are defined. In a second step, the artifact types are specified.

4.1 Requirements for the alignment architecture and the artifact types

The goal is to decouple business architectures and IT architectures so that changes to business-related artifacts do not necessarily have to be propagated into IT-related artifacts and vice versa. The conceptual model for the structure of such a mechanism is called alignment architecture. This is the focus of the following analysis. Furthermore, the implicit behavior of the alignment architecture and the additional explicit behavioral properties of the mapping mechanism are to be described. These aspects are a subject for further research activities on alignment architectures.

Alignment artifact types are needed so that existing direct relationships between business architecture and IT architecture can be replaced by

1. m:1 relationships between business artifacts and artifacts of the alignment architecture and
2. 1:n relationships between artifacts of the alignment architecture and IT artifacts.

By establishing m:1 and 1:n relationships, a difference in complexity between the align-

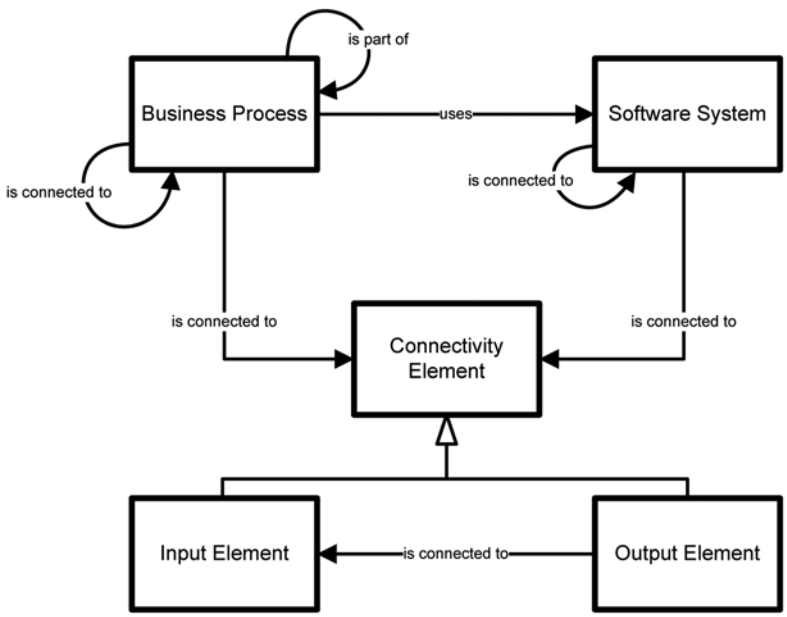


Fig. 4 Meta model of the modeling approach

ment layer and adjoining layers is needed in order to intercept the interdependencies between business architecture and IT architecture. As in the case of aggregation levels for business architecture and IT architecture, aggregation levels must also be considered for the alignment layer (Stünzer 1996, p. 73).

In analogy to the examples introduced above, the proposed alignment architecture and its artifacts primarily serve as a translator between the linked structures – they belong neither to business architecture nor to IT architecture.

4.2 Artifacts of the alignment architecture

Based on the requirements summarized above, the artifact types enterprise service, application and domain are proposed. These artifact types are in hierarchical relationship to each other (cf. **Fig. 2**). The relationship can but does not need to be strictly hierarchical, and therefore might be poly-hierarchical.

Enterprise services represent elementary bundles of business functionality, arranged due to collective support of business processes, usage of same information objects or collective reuse (Schelp and Winter 2008). Each bundle is implemented by one or more software components (1: n relationship) and can be used in one or more business processes (m:1 relationship). Applications group enterprise services to collectively support business processes, use the same information objects or are collectively reused (Winter 2003). Domains are groups of applications (and hence groups of enterprise services) which support coherent business processes, or use coherent software systems, or use coherent information objects and therefore form a designated integration area (Hagen 2003).

In analogy to **Fig. 1, 2** illustrates the alignment architecture for IT/business alignment. As in systems theory, computer science and economics, the decoupling layer is placed ‘in between’ the business and the IT layer. The alignment layer only serves as a translator between the linked structures and belongs neither to business architecture nor to IT architecture. It decouples artifacts of different types. Direct m:n relationships are dissolved into m:1 and 1:n relationships. Artifacts of the decoupled layers can often be modified without necessarily having to propagate the modifications into the other layer. It can be expected that the contribution of the alignment architecture will decrease as the specificity of the artifacts of the alignment architecture grows. Therefore, alignment artifacts should be more aggregated than the decoupled artifacts.

5 Construction specifications for artifacts of the alignment architecture

As a prerequisite for the systematic development of the alignment architecture, enterprise services, applications and domains have to be identified, enterprise services need to be reasonably grouped into applications and applications assigned to domains. Approaches to enterprise service design have been proposed only recently (Heutschi and Legner 2007; Schelp and Winter 2007). These abstract design guidelines are composed on the basis of the respective goals. Therefore, the design of enterprise services can be data-oriented, process-oriented, or reuse-oriented (Schelp and Winter 2008).

Unlike enterprise service design, research and practice of application design has a long history (IMG 1999; Winter 2003).

Based on approaches from the 1960s, IBM’s Business Systems Planning Method (BSP, IBM 1984) found worldwide diffusion in the 1980s. BSP aims to group IT functionalities according to data use and thereby identify suitable applications. Alternatively, IMG propose the Systems and Technology Planning Method (STP, IMG 1999) for application design. In this case IT functionalities are grouped according to ‘similar’ responsibilities instead of ‘similar’ data usage (**Fig. 3**).

The systematical definition of domains has rarely been the subject of scientific contributions in information systems research. Practitioner approaches have been published by (Bath and Herr 2004; Hagen 2003). Similar to enterprise service design, domains are built according to conceptual guidelines (e. g. “minimize interdependencies of elements in different domains and maximize interdependencies of elements within the same domain”) using a top-down approach.

Existing construction guidelines for enterprise services and domains usually use top-down approaches. BSP does follow a bottom-up direction, but addresses only technical artifacts (such as IT functionalities and data structures) and is therefore not suitable for IT/business alignment. STP also uses a bottom-up approach and does include technical as well as business-related artifacts (IT functionalities and organizational responsibilities), but does not consider data coupling or reuse in the proposed application design approach. If business-related and IT artifacts were analyzed and more than one coupling dimension was considered, this bottom-up approach would be suitable to design coarse-grained alignment artifacts.

Schelp and Winter (2008) use empirical research to show that design principles for enterprise services and applications are the same. Based on these findings, it can be assumed that the differentiation between enterprise services, applications and domains is related to the level of aggregation and not different design principles. In a multilevel hierarchy, domains can be subdivided into applications which can be further decomposed into enterprise services. The borders of these hierarchical levels depend on the aggregation.

We propose a bottom-up analysis of the relationships between business architecture and IT architecture to design alignment artifacts. Depending on the level of aggregation of the relationships analyzed, the result will be enterprise services, applications or domains.

The following section presents a bottom-up approach to identify alignment artifacts using the example of domain clustering.

6 An explorative bottom-up approach for the identification of alignment artifacts using the example of domain clustering

In the following section a formal clustering algorithm is applied to identify domains on the basis of organizational architecture and software architecture. In order to use such an algorithm, a model of organizational architecture and software architecture is needed. First, the underlying meta model is presented. Afterwards, the clustering algorithm is introduced and the prototype described.

6.1 Proposed meta model

Our modeling approach focuses on the representation of links between the business layer and the IT layer. The respective meta model considers the following elements in a simplified manner:

- Business processes on a high level of aggregation
- Software systems on a high level of aggregation
- Relations which reflect the usage of a software system to support a business process
- Interrelations and interfaces between software systems which support a business process

In order to enable consistent modeling on any desired level of detail, input and output elements are introduced as connectivity elements (**Fig. 4**).[2] To enable application of the clustering algorithm, the instance of the meta model has to be consistently modeled at the desired level of detail. It is especially crucial that there are no 'gaps' in the model, which would render the clustering results useless.

6.2 Formalization of domain building

Instances of the alignment architecture meta model can be transformed into a graph. In the following sections we will give a short introduction to graph theory and algorithms for graph partitioning. Thereafter we will introduce our implementation of these algorithms for clustering business-related and IT-related artifacts. Since the analysis takes place on an aggregated level of detail, the resulting clusters constitute domain candidates on the alignment layer.

A *graph* consists of vertices V and edges E. All elements of our model (business processes and software systems) can be considered vertices and their relations can be considered edges. The alignment architecture model can be transformed into a graph. All structurally relevant elements (business processes, software systems) can be represented as vertices and their connections (usage of software systems in business processes, control flows, software system interfaces) can be represented as edges. Graphs which are attributed by numerical values are called *weighted graphs*. Edges with a defined start and end are called *directed edges*. If this is not the case, the edges are called *undirected edges*. The *distance* between two vertices is defined as the shortest path between them. In the case of weighted graphs this distance is called *weighted distance* (O'Madadhain et al. 2005, p. 6). A graph with n vertices is represented by an $n \times n$ adjacency matrix A with elements:

$$a_{ij} = \begin{cases} 1 & \text{wenn die Knoten } i \text{ und } j \text{ verbunden sind} \\ 0 & \text{sonst} \end{cases}$$

In a weighted graph a_{ij} represents the weight of the edge that links the vertices i and j. The degree k of a vertex i is defined as the number of edges that are linked to this vertex:

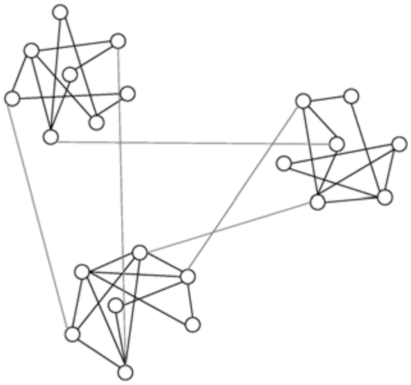


Fig. 5 Social system with a community structure (Girvan and Newman 2002, p. 7822)

$$k_i = \sum_j a_{ij}$$

The existing models can be transformed into such graphs. In the first instance, the type to which the elements comply is irrelevant. They are represented as coequal vertices. The edges are adopted in the graph as undirected edges because direction information is not significant for the structural interrelations. The weights of these edges are derived from the number of edges connecting two vertices. A weight greater than one applies when two connected software systems are used in different processes.

As the most important criteria for domain building, the minimization of interdependencies between different domains and the maximization of interdependencies between elements within one domain (meaning the shifting of elements into a domain) can be extracted from the examples in section 3 as well as from the requirements in section 4.1. Even when considering small extracts from business and IT architectures, the application of such an instruction will soon become unmanageable without the appropriate tool support. We therefore identified and applied algorithms which evaluate the togetherness of different elements from the structure of a graph. The semantics of the model elements are ignored, and only the model structure is evaluated. According to the analysis approach, all elements which are closely linked should form one domain.

Such problems are discussed using the key words graph *partitioning* or *cluster-*

[2] Connectivity elements primarily support consistent modeling. They ensure that continuous graphs can be drawn across various levels of detail within a model.

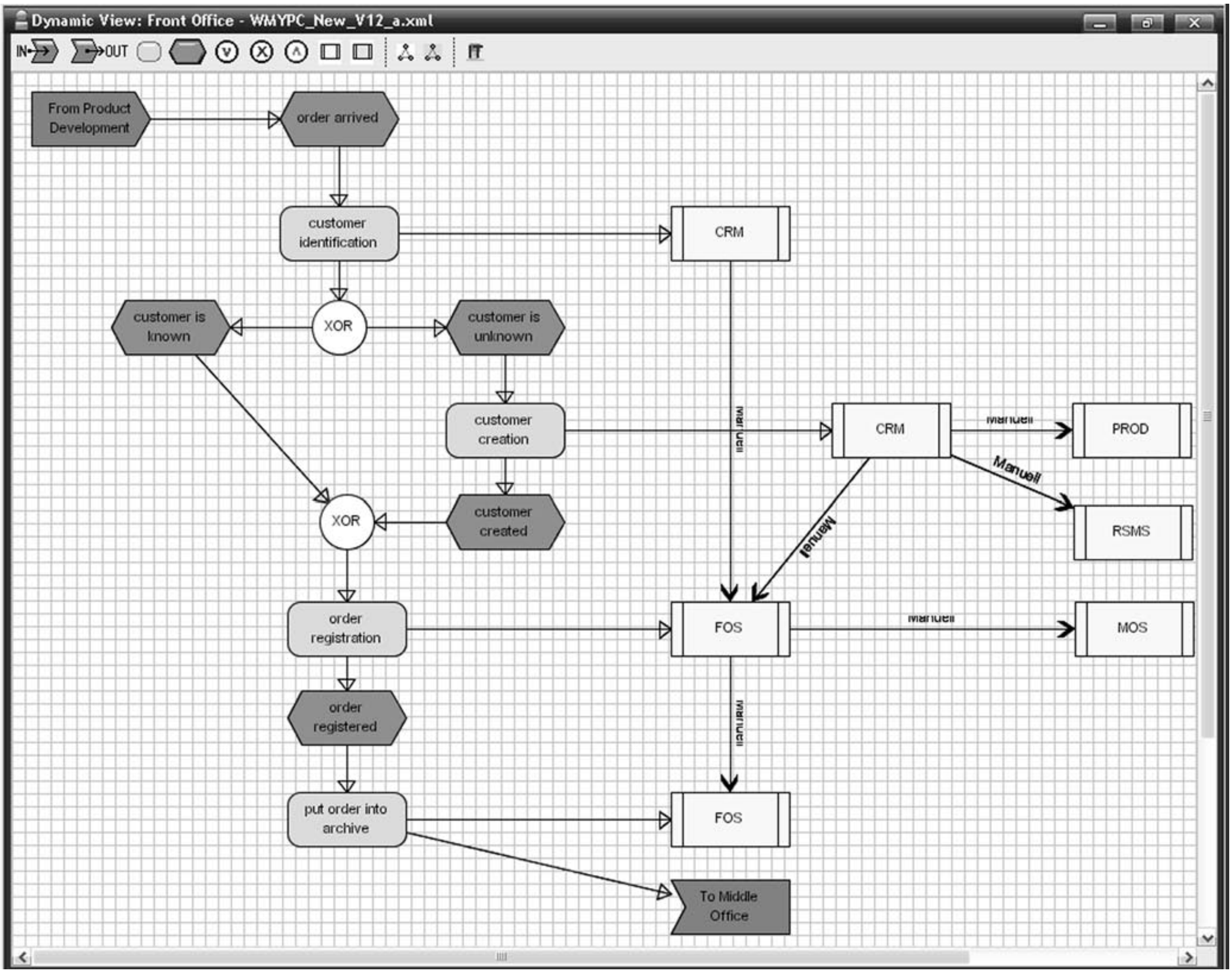


Fig. 6 Modeling in the prototype (part of the model)

ing. A cluster represents a set of elements which are similar in one form or another (O'Madadhain et al. 2005, p. 18). In the case at hand, a similarity emerges from the fact that different elements share a subset of neighbors.

Girvan and Newman (2002, p. 7821) developed a clustering algorithm to determine communities within social systems. Such a system consists of a set of individuals (vertices) that have interrelations (edges). For example, a relation is assumed if two people know each other. How can communities be identified in this setting? **Fig. 5** represents such a system where clusters (communities) have been emphasized. Like Girvan and Newman, we have analyzed existing clustering algorithms in terms of their performance to detect clusters within graphs based on their structure.

In certain constellations (especially with vertices at the border of a graph which are linked to the graph by only one edge), these algorithms deliver poor results: Elements have been excluded from a cluster although they should be part of it. Other algorithms, such as the Voltage Clusterer (Wu and Huberman 2004) have also been tested, but did not produce useful results.

The "Betweenness" algorithm by Girvan and Newman does not try to find the central edges of a graph. Instead, it focuses on the identification of edges which are "least central" and therefore lie "most between" communities. In the past, this vertex betweenness has been studied as measure of the influence of one vertex on the graph. Proposed firstly by Freeman (1977), the betweenness-centrality of one vertex *i* is defined as the number of shortest paths between pairs of other vertices which run across it. Girvan and Newman generalize Freeman's betweenness-centrality and define the "edge-betweenness" of an edge as the number of shortest paths between pairs of vertices which run across this edge. If there are communities in a graph, these are interlinked by only few edges. That means that the shortest paths between these communities run across these few edges. Therefore, these edges will have a high edge-betweenness. By removing these edges, the communities can be separated and the underlying structure is revealed.

The so-called edge-remover algorithm for weighted graphs is to be applied as follows (Newman 2004, p. 4):

1. Calculate the betweenness for all edges in the network.
2. Divide the betweenness by the weight of the respective edge.
3. Remove the edge with the highest resulting betweenness.
4. Recalculate the betweenness for all remaining edges.
5. Repeat steps 3 and 4 until no edges are left.

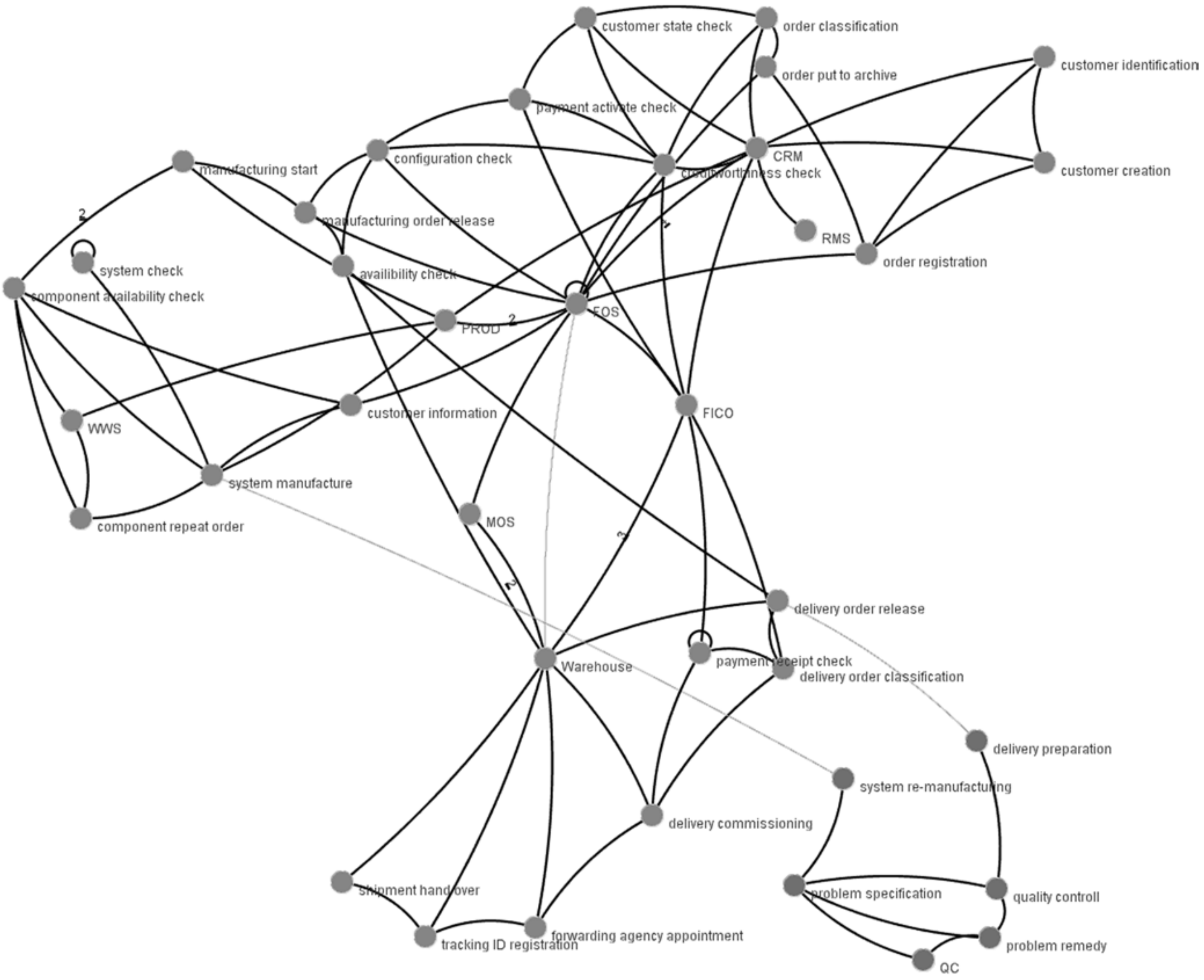


Fig. 7 Model transformed into a graph

A further question is when a *good clustering* is achieved by removing edges or when further edges should be removed elsewise. Newman (2004, p. 7) defines the *modularity Q* for this purpose. He calculates the fraction of edges in a graph which are located within communities as follows:

$$\frac{\sum_{ij} a_{ij}\delta(c_i,c_j)}{\sum_{ij} a_{ij}} = \frac{1}{2m}\sum_{ij} a_{ij}\delta(c_i,c_j)$$

Where *c* is the community to which the vertex *i* belongs. The function *δ*(*u*,*v*) is 1 if *u*=*v* and 0 otherwise.

$m = \frac{1}{2}\sum_{ij} a_{ij}$ is the number of edges in the graph. If the degree k_i of all vertices *i* is preserved but edges are spread at random in the network, the probability of an edge existing between vertices *i* and *j* is $k_i k_j/2m$. Thus the modularity *Q* is given by

$$Q = \frac{1}{2m}\sum_{ij}\left[a_{ij} - \frac{k_i k_j}{2m}\right]\delta(c_i,c_j).$$

Values for *Q* range between 0 and 1. A value of 0 indicates that there will be no edges left in a community after a clustering, which would be expected according to the random allocation. Values for *Q* indicating a good clustering range between 0.3 and 0.7.

6.3 Prototypic implementation

The modeling and clustering approach for the presented context has been implemented in a prototype.

Fig. 6 shows part of a model that has been used in the prototype. The business processes have been represented as event-driven process chains (EPC). In addition, the usage of software systems along the business processes, the interfaces between the software systems along the process flow, and the inputs and outputs for the hierarchical refinement of the models are represented.

The performance of the clustering algorithm used has been analyzed in a sequence of special test scenarios (Aier 2006). Prior to the discussion on the application of domain clustering by the means of a case study in the following chapter, the functionality will be demonstrated in a simple test scenario.

Fig. 7 shows the model belonging to the test scenario which has been transformed into a graph.[3] With 20 edges being removed, the modularity function reaches its maximum. After the application of the clustering algorithm and the removal of 20 edges (**Fig. 8**) the graph is separated into five clusters. **Fig. 8** shows the resulting

[3] Events and operators were eliminated while transforming the model into a graph since they have no impact on the structure of an undirected graph.

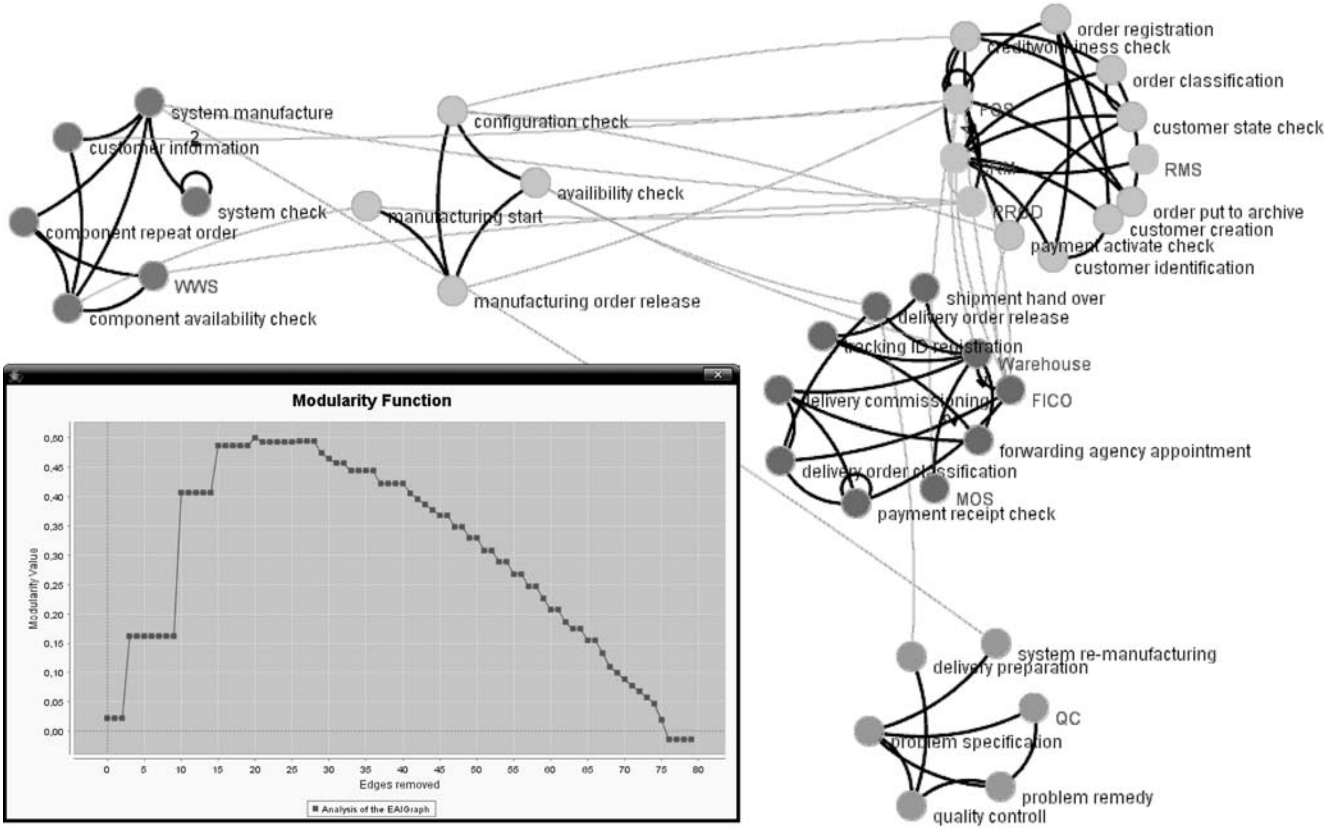


Fig. 8 Model transformed into a graph after the removal of 20 edges

clusters in a grouped illustration as well as the progress of the respective modularity function over the number of removed edges.

The presented approach aims at deducing a domain architecture on the alignment layer from the existing software and process structures by applying a bottom-up approach. For this purpose it is important to identify the central coupling elements. The clustering algorithm detects candidates for the domains on the alignment layer. A cluster equals an element onto which m elements of the business architecture and n elements of the IT architecture can be mapped. Thereby, as many elements as possible are mapped onto one cluster – without losing too much specificity.

6.4 Case study of a telecommunications supplier

In order to evaluate the performance of the proposed approach beyond experiments with test models, an extensive case study was conducted. In the following section, the case is described first. Afterwards, the results of the evaluation are presented and discussed.

6.4.1 Company and Scenario

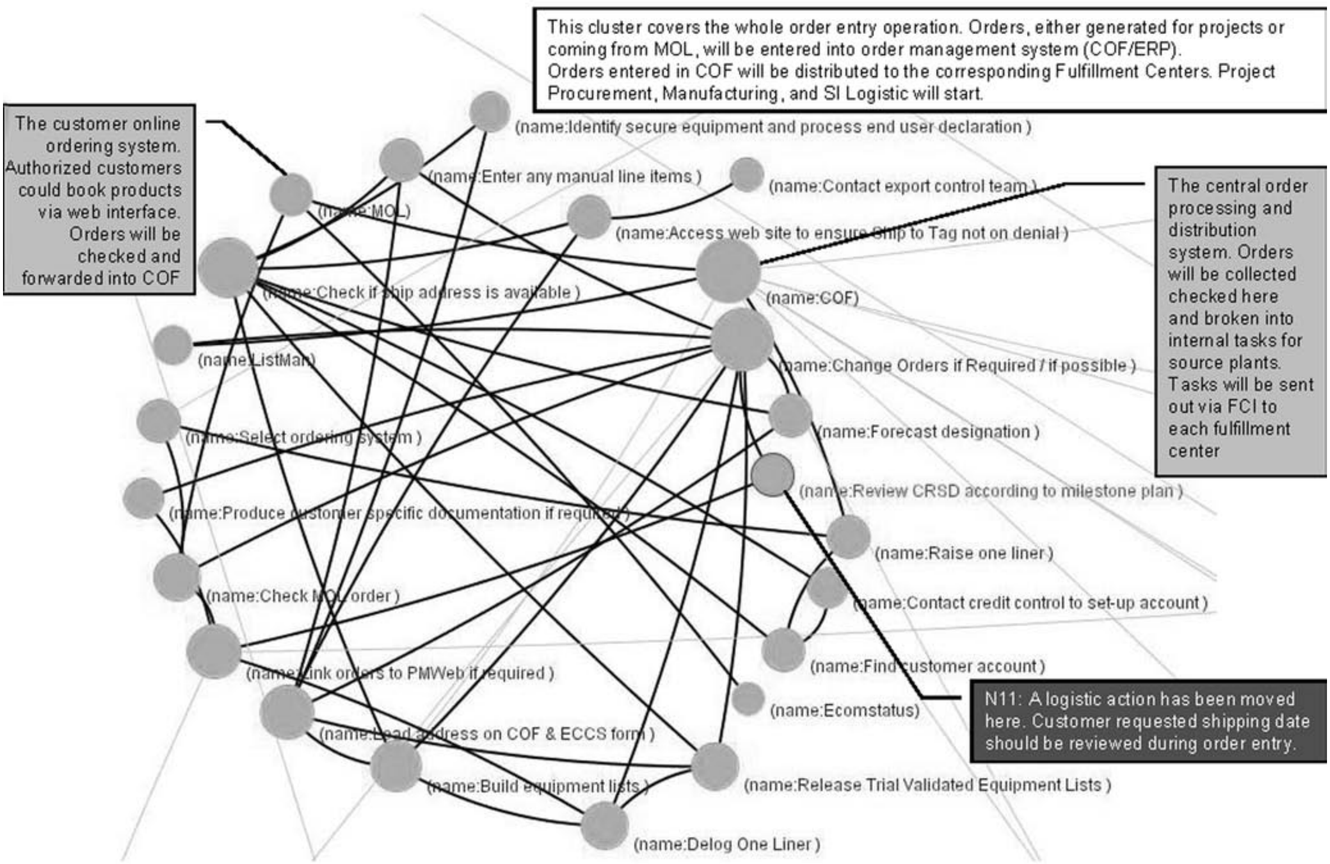
The study was conducted at a US telecommunications supplier that operates worldwide. The company belongs to the Fortune-100 and accounted for revenues of 36.6 billion US dollars in 2007. The business unit considered produces, configures and sells professional radio systems to corporate clients and public sector clients worldwide.

The starting point for the case study was a project which aimed at aligning the complete software system landscape of the business unit in focus with the respective business structures. The case study set out to identify the latently existing structure of the totality of software systems and business processes in the form of domain models. Based on the domain identification, a redesign of the IT architecture was planned for those domains with a particularly high need for action. This was to lead to an efficient decoupling of independent artifacts and a stable IT/business alignment. The business area considered comprises about 100 software system components and about 400 activities – as well as the respective events, operators and relations in the process model.

The business process in focus spans unstructured market observation in preparation for a bidding process, the construction of the radio system at the integration center, the installation of the radio system at distributed locations, the operation of the system for about 25 years, and the subsequent disassembly and disposal of the radio system. As a first step, the business processes, the software system landscape and the interrelations between them were investigated by means of the inventory method and interviews. The findings were documented as an instance of the above proposed meta model and modeled with the prototype (Frank et al. 2007, pp. 152–159, 165–168).

6.4.2 Results

By applying the clustering algorithm to the model, 44 domains were identified. **Fig. 9** illustrates one of these domains as an example. The domains were named in the subsequent manual analysis, and 24 striking characteristics were documented in the cluster annotation report. For example, domains without IT support were noted, or elements were noted which were assigned to a certain domain but expected


This cluster covers the whole order entry operation. Orders, either generated for projects or coming from MOL, will be entered into order management system (COF/ERP). Orders entered in COF will be distributed to the corresponding Fulfillment Centers. Project Procurement, Manufacturing, and SI Logistic will start.

The customer online ordering system. Authorized customers could book products via web interface. Orders will be checked and forwarded into COF

The central order processing and distribution system. Orders will be collected checked here and broken into internal tasks for source plants. Tasks will be sent out via FCI to each fulfillment center

N11: A logistic action has been moved here. Customer requested shipping date should be reviewed during order entry.

(name:Identify secure equipment and process end user declaration)
(name:Enter any manual line items)
(name:Contact export control team)
(name:MOL)
(name:Access web site to ensure Ship to Tag not on denial)
(name:Check if ship address is available)
(name:COF)
(name:ListMan)
(name:Change Orders if Required / if possible)
(name:Select ordering system)
(name:Forecast designation)
(name:Produce customer specific documentation if required)
(name:Review CRSD according to milestone plan)
(name:Raise one liner)
(name:Check MOL order)
(name:Contact credit control to set-up account)
(name:Find customer account)
(name:Link orders to PMWeb if required)
(name:Ecomstatus)
(name:Load address on COF & ECCS form)
(name:Build equipment lists)
(name:Release Trial Validated Equipment Lists)
(name:Delog One Liner)

Fig. 9 Domain order management in the rollout of a radio system

in another domain. Amongst these, there were six misassignments from a business perspective, nine obvious improvement potentials, and nine candidates for further discussion (e. g. expected business process changes or future IT support potentials).

The discussion of the clustering results facilitated an understanding of the latently existing structures among the stakeholders. It was a good exercise for understanding the interplay of software systems and processes as well as for detecting and understanding IT support gaps.

Finally, domain building was discussed with the respective process owners' IT representatives. As a result, it could be stated that the originated clusters were reasonable from a business perspective. Nevertheless, the clusters were partly surprising because they did not always reflect the officially specified organizational groups and competencies. However, it cannot be concluded that the organizational structure should be changed according to the clustering results. According to Frese (1993, p. 1021), these entities/domains are no blueprints for restructuring large enterprises, but instead secondary structures spanning multiple business areas which act as a role model for the principles of "clarity" and "seclusion". In our context they represent coarse-grained alignment artifacts for integrating business processes and software systems by linkage. The alignment artifacts serve to decouple business architectures from IT architectures.

The identified domains represent artifacts of the alignment architecture which decouple artifacts of the adjoining architectural layers. Between the domains, it is necessary to check if the edges "deleted" by the domain-building process can also be consolidated in order to achieve the goal of loose coupling. The number of edges removed per cluster can be understood as a quality measure for business and IT architectures. In particular, software systems which have many edges to elements of other domains (e. g. the software system "COF" in **Fig. 9**) should be split up into different services on a more detailed level. This bears the biggest potential for further decoupling of architecture layers: Changes which affect only parts of the current "COF component" remain local.

A limiting factor of this approach can be seen in the strong dependence on careful and consistent modeling, since the model quality will have significant impact on the clustering results. At the same time, the approach delivers benefits by supporting analysis of business and IT architectures and designing the alignment architecture. Even in the case study scenario, the complexity of the structures was not intuitively manageable without the use of the algorithm.

7 Discussion: conclusion and outlook

This article focuses on the integration of business architectures and IT architectures, which is considered to be one of the core issues in design-oriented information systems research. Based on decoupling mechanisms found in systems theory and application examples from computer science and economics, virtual, coarse-grained mapping architectures are described and analyzed. An alignment architecture is proposed which replaces direct m:n relationships between business architecture and IT architecture by a linking mechanism. Therefore, changes to either architecture do not necessarily cause changes to the other.

Compared to existing approaches, the contribution of the presented approach can be seen in the following aspects:

- Classical middleware, enterprise application integration and other m:n mechanisms in the IT domain are limited to link artifacts of the same kind. Only a new architectural layer with dedicated mapping artifacts can decouple artifacts of business architecture and IT architecture.
- Even service-oriented architectures can increase flexibility by decoupling, but also focus on the link between the same artifact types. Service orientation on the business and IT side cannot solve the problem outlined in the first section. A systematic approach to align both architectures is necessary. Therefore alignment artifacts should be designed and structured in a service-oriented way. This is also important to enable consistent connections between artifacts that are preferably service-oriented too.
- Reference architectures for information systems (such as Retail-H or VAA) represent proven and tested solutions that serve as a reference and are reusable to a certain extent. Alignment issues are not addressed in reference architectures because reference models represent one point in time and not a transition process. IT/business alignment is an important part of information management because the desirable but complex transformation of an architecture (e. g. based on a reference model) to achieve consistency between business and IT can easily be lost due to frequent changes on both sides. The alignment architecture presented in this paper addresses these dynamic aspects. It does not gain any benefits in a stable environment.

As mentioned in the introduction to this article, our research process follows a common model for information system design research (Rossi and Sein 2003). This process requires the evaluation of proposed artifacts. Therefore, after the proposition of bottom-up design of virtual alignment artifacts, the approach was implemented using the example of domain clustering and a case study was presented.

Future research is needed to integrate design approaches for the various alignment layer artifacts. Such methodology could (as shown above) follow a bottom-up approach, i. e. could be based on existing business-related and IT-related architectures. Alternatively, it seems possible and vital for checking consistency to deduct alignment artifacts from the high-level to-be business architecture and IT architecture. The particular challenge seems to lie in harmonizing the different design goals of business architecture (process efficiency and process effectiveness), IT architecture (reusability, scalability, performance, etc.) with the alignment goal of the proposed new layer. Then again, the results of a model-driven bottom-up approach can be used as a further perspective in top-down activities. The presented bottom-up approach is based on as-is business and IT architectures. The “buffer” effect of the alignment architecture can be amplified by implementing “loose coupling” within the decoupled layers. Further research activities should investigate how bottom-up analysis of as-is architectures can support the design of to-be structures. The number of reduced edges per cluster can in this case serve as an evaluation measure.

While the decoupling characteristics of the alignment architecture do not imply consistency of the architectural layers, the hierarchical structure of artifacts nevertheless needs design policies. Furthermore, the question of whether only a combination of 1:n and m:1 relationships delivers efficient results, or if there are scenarios where directed m:n linkages lead to better results should be analyzed in detail.

So far the analysis of mapping mechanisms focuses on structural aspects. Further research is required to investigate the interfacing behavior of the artifacts of business, IT and alignment architecture.

Furthermore, an “infrastructure” for development and operation of an alignment architecture needs to be developed. This might include repositories for storing relevant parts of business and IT architectures. Such model repositories create the necessary transparency to develop the alignment architecture and to advance business and IT architectures.

At the same time, the achieved transparency is prerequisite to monitoring the effectiveness and efficiency of this architectural approach over time. It would then be possible to measure e. g. what change frequency and what degree of complexity would jus-

Abstract

Stephan Aier, Robert Winter

Virtual Decoupling for IT/Business Alignment – Conceptual Foundations, Architecture Design and Implementation Example

IT/business alignment is one of the main topics of information systems research. If IT artifacts and business-related artifacts are coupled point-to-point, however, complex architectures become unmanageable over time. In computer science, concepts like the ANSI/SPARC three-level database architecture propose an architecture layer which decouples external views on data and the implementation view of data. In this paper, a similar approach for IT/business alignment is proposed. The proposed alignment architecture is populated by enterprise services as elementary artifacts. Enterprise services link software components and process activities. They are aggregated into applications and subsequently into domains for planning/design and communication purposes. Most design approaches for the construction of enterprise services, applications and domains are top-down, i. e. they decompose complex artifacts on a stepwise basis. As an alternative which takes into account coupling semantics, we propose a bottom-up approach which is demonstrated for the identification of domains. Our approach is evaluated using a telecommunications equipment case study.

Keywords: Integration, Decoupling, IT/business alignment

tify the effort involved in introducing an additional alignment architecture.

References

Aier S (2006) How clustering enterprise architectures helps to design service oriented architectures. In: Proceedings of the IEEE international conference on services computing (SCC'06). IEEE Computer Society, Los Alamitos, pp 269–272

Ashby WR (1981) The set theory of mechanism and homeostasis. In: Conant R (ed) Mechanisms of intelligence: Ross Ashby's Writings on Cybernetics. Intersystems, Seaside, pp 21–49

Bath U, Herr M (2004) Implementation of a service oriented architecture at Deutsche Post MAIL. In: Aier S, Schönherr M (eds) Enterprise Application Integration – Serviceorientierung und nachhaltige Architekturen. GITO, Berlin, pp 279–297

Buxmann P, König W (1998) Das Standardisierungsproblem: Zur ökonomischen Auswahl von Standards in Informationssystemen. WIRTSCHAFTSINFORMATIK 40(2):122–129

Buxmann P, Weitzel T, von Westarp F, König W (1999) The standardization problem: an economic analysis of standards in information systems. In: Proceedings of the 1st IEEE conference on standardisation and innovation in information technology SIIT '99, pp 157–162

Date CJ (2000) An introduction to database systems, 7. edn. Addison Wesley, Reading

Devlin B (1997) Data warehouse: from architecture to implementation. Addison Wesley, Reading

Dietrich J (2008) Nutzung von Modelliersprachen und -methodologien standardisierter B2B-Architekturen für die Integration unternehmensinterner Geschäftsprozesse. Gito, Berlin

Frank H, Bobrik A, Haarländer N (2007) Vorgehensmodell der Systemanalyse. In: Krallmann H, Schönherr M, Trier M (eds) Systemanalyse im Unternehmen – Prozessorientierte Gestaltung von Unternehmen, 5th edn. Oldenbourg, München, pp 135–186

Freeman LC (1977) A set of measures of centrality based upon betweenness. Sociometry 40(1):35–41

Frese E (1993) Geschäftssegmentierung als organisatorisches Konzept. ZfbF 45(12):999–1024

Fritsch M, Wein T, Ewers H-J (2005) Marktversagen und Wirtschaftspolitik: mikroökonomische Grundlagen staatlichen Handelns, 6th edn. Vahlen, München

Girvan M, Newman MEJ (2002) Community structure in social and biological networks. Proceedings of the National Academy of Science 99(12):7821–7826

Glassman RB (1973) Persistence and loose coupling in living systems. Behavioral Science 18(2):83–98

Hagen C (2003) Integrationsarchitektur der Credit Suisse. In: Aier S, Schönherr M (eds) Enterprise Application Integration – Flexibilisierung komplexer Unternehmensarchitekturen. GITO, Berlin, pp 61–81

Henderson JC, Venkatraman N (1993) Strategic alignment: leveraging information technology for transforming organizations. IBM Systems Journal 32(1):4–16

Herdzina K (2005) Einführung in die Mikroökonomik, 10th edn. Vahlen, München

Heutschi R, Legner C (2007) SOA adoption in practice – findings from early SOA implementations. In: Proceedings of the 15th European conference on information systems "relevant rigour – rigorous relevance"

IBM (1984) Business systems planning – information systems planning guide, IBM-Form GE20-0527-4, 4th edn., IBM, Atlanta

IMG (1999) PROMET STP: Methodenhandbuch für die System- und Technologieplanung, Release 1.0. IMG AG, St. Gallen

Inmon WH (1996) Building the data warehouse, 2th edn. Wiley Computer Publishing, New York

Kaib M (2002) Enterprise Application Integration – Grundlagen, Integrationsprodukte, Anwendungsbeispiele. DUV, Wiesbaden

Kantzenbach E (1967) Die Funktionsfähigkeit des Wettbewerbs, 2th edn. Vandenhoeck & Ruprecht, Göttingen

Katz ML, Shapiro C (1994) Systems competition and network effects. J Economic Perspectives 8(2):93–115

Kerber W (2007) Wettbewerbspolitik. In: Bender D, Berg H, Cassel D et al (eds) Vahlens Kompendium der Wirtschaftstheorie und Wirtschaftspolitik, Vol. 2. Vahlen, München, pp 369–434

Kollmann T (2001) Virtuelle Marktplätze: Grundlagen – Management – Fallstudie. Vahlen, München

Linthicum DS (2000) Enterprise application integration. AWL Direct Sales, Reading

Luftman JN (2005) Key issues for IT executives 2004. In: MIS Quarterly Executive 4(2):269–285

Luhmann N (1991) Soziologie des Risikos. de Gruyter, Berlin

Luhmann N (2002) Einführung in die Systemtheorie. Carl Auer, Heidelberg

Meffert JPH (1993) Standards als Integrationsinstrument in der Computer- und Kommunikationsindustrie: wettbewerbsstrategische Bedeutung und Durchsetzung. Universitätsverlag, Konstanz

Mertens P (2007) Integrierte Informationsverarbeitung 1, 16th edn. Gabler, Wiesbaden

Newman MEJ (2004) Analysis of weighted networks. Phys Rev E 70 (056131)

O'Madadhain J, Fisher D, Smyth P, White S, Boey Y-B (2005) Analysis and visualization of network data using JUNG. J Statistical Software (Preprint)

Perrow C (1999) Normal accidents: living with high-risk technologies. Princeton University Press, Princeton

Ricardo D (1830) On the principles of political economy and taxation, 2th edn. Bell, Washington

Rosemann M (1999) Gegenstand und Aufgaben des Integrationsmanagements. In: Scheer A-W, Rosemann M, Schütte R (eds) Integrationsmanagement. Institut für Wirtschaftsinformatik, Universität Münster, pp 5–18

Rossi M, Sein MK (2003) Design research workshop: a proactive research approach. http://www.cis.gsu.edu/~emonod/epistemology/Sein%20and%20Rossi%20-%20design%20research%20-%20IRIS.pdf. Accessed 2008-08-14

Schelp J, Winter R (2007) Towards a methodology for service construction. In: Proceedings of the 40th Hawaii international conference on systems sciences (HICSS-40). IEEE Computer Society, Los Alamitos, p 64a

Schelp J, Winter R (2008) Entwurf von Anwendungssystemen und Entwurf von Enterprise Services – Ähnlichkeiten und Unterschiede. WIRTSCHAFTSINFORMATIK 50(1):6–15

Shapiro C (2001) Setting compatibility standards: cooperation or collusion? In: Cooper Dreyfuss R, Leenheer Zimmerman DL, First H (eds) Expanding the bounds of intellectual property. University Press, Oxford, pp 81–101

Smith A (1776) An inquiry into the nature and causes of the wealth of nations. Strahan and Cadell, London

Stünzer L (1996) Systemtheorie und betriebswirtschaftliche Organisationsforschung. Duncker & Humblot, Berlin

Vogler P (2003) Prozess- und Systemintegration: Umsetzung des organisatorischen Wandels in Prozessen und Informationssystemen. Habilitation, Universität St. Gallen

Weiber R, Kollmann T (1998) Competitive advantages in virtual markets – perspectives of "information-based marketing" in cyberspace. Eur J Marketing 32(7/8):603–615

Weick KE (1976) Educational organizations as loosely coupled systems. Administrative Science Quarterly 21(1):1–19

Winter R (2003) An architecture model for supporting application integration decisions. In: Ciborra C et al (eds) Proc. 11th European conference on information systems

Winter R (2008) Business Engineering – Betriebswirtschaftliche Konstruktionslehre und ihre Anwendung in der Informationslogistik. In: Dinter B, Winter R (eds) Integrierte Informationslogistik. Springer, Heidelberg, pp 17–38

Winter R, Schelp J (2006) Dienstorientierte Architekturgestaltung auf unterschiedlichen Abstraktionsebenen. In: Reussner R, Hasselbring W (eds) Handbuch der Software-Architektur. dpunkt, Heidelberg, pp 229–242

Wu F, Huberman BA (2004) Discovering communities in linear time: a physics approach. Eur Phys J B 38(2):331–338