Model Driven Service Domain Analysis

Conference Paper · December 2006

DOI: 10.1007/978-3-540-75492-3_17

CITATIONS
13

READS
288

2 authors:

Stephan Aier
University of St.Gallen
184 PUBLICATIONS **2,086** CITATIONS

SEE PROFILE

Marten Schönherr
Automat Berlin
60 PUBLICATIONS **635** CITATIONS

SEE PROFILE

Model Driven Service Domain Analysis

Stephan Aier[1] and Marten Schönherr[2]

[1] Institute of Information Management, University of St.Gallen
Mueller-Friedberg-Strasse 8, 9000 St. Gallen, Switzerland
stephan.aier@unisg.ch
[2] Faculty of Computer Science and Electrical Engineering,
Technical University Berlin,
Franklinstr.28/29, 10587 Berlin, Germany
mschoenherr@sysedv.tu-berlin.de

Abstract. Currently scientists and practitioners are discussing Service Oriented Architectures (SOA) as an approach to reconcile business requirements and IT. The alignment of business and technology in organizations is a key challenge in the discipline of Enterprise Architecture (EA). Therefore the contribution starts with a discussion of SOA as an EA integration concept to synchronize business requirements and IT architecture in an efficient way. Differentiating methodological and technological aspects of EA the paper shows the need for methods in the field of domain analysis supporting the design of a SOA. The main contribution of the paper is an algorithm based modeling tool and methodology to support service domain clustering. Service clusters are being used for service definition and management. Due to enormous complexity it is necessary to support architects by finding and defining appropriate clusters. For modeling interdependencies in EA the paper's focus is on business processes, information systems and interfaces. Our approach adopts network-centric algorithms used in the field of social network analysis to define and/or identify service domain clusters in complex scenarios. Edge remover algorithm is used to compute the relevant model aspects. The results of our approach will be demonstrated in a case study.

Keywords: SOA, Enterprise Architecture, Process Oriented Integration, Service Domain Clustering.

1 Challenges of Managing Complex Enterprise Architectures

During the last decades IT has been grown a determining success factor. As a result especially in large organizations existing IT infrastructures can be described as extremely complex and heterogeneous. Therefore interoperability is one of the main issues developing and operating these infrastructures. As a matter of fact the conventional way of using individually coded point-to-point interfaces to connect systems is getting beyond control due to increasing overall system complexity. On the one hand one has to consider the costs for IT operation and maintenance of up to thousands of individual interfaces and on the other hand adaptations caused by the introduction of new systems (or system upgrades) and/or procedural-organizational stipulations are

D. Georgakopoulos et al. (Eds.): ICSOC 2006 Ws, LNCS 4652, pp. 190–200, 2007.

almost non-manageable due to complexity of element interdependencies. The recently discussed prominent approach to this issue is a Service Oriented Architecture (SOA). The paradigm of SOA is a distributed integration infrastructure [1]. One of the key benefits is a major challenge as well—SOA's integration level. SOA not only aims at systems integration at a primarily technical level, but on process integration driven by business requirements and implemented by utilizing information technology.

Such an integration approach inevitably affects organizational as well as technological questions. In an empirical study we found out, that in companies using integration technologies as SOA, a third of their applications are integrated utilizing those integration platforms and a further third are planned to be integrated this way in the near future [2]. To sum it up it can be said that integration technologies as SOA are major elements of enterprise architectures (EA).

The definition of EA is a central part of this paper. The term of (enterprise) architecture is used in multiple meanings and suffers from a lack of consistent definition appropriate to specific research domains as Business Informatics, Computer Science or Management Science. Therefore in the following we will describe our understanding of enterprise architecture and the role of integration concepts in the context given.

In a few words an architecture can be defined as an abstract and holistic concept of structures and patterns considering planning aspects [3]. Architectures are generally results of planning efforts and offer by definition a master plan supporting holistic implementation for future actions. These universal characteristics can be used for planning and designing of enterprise structures and strategies too. Furthermore an enterprise architecture considers organizational, technical and psychosocial aspects for planning and building Information Systems (IS) in a socio-technical manner. This contribution particularly focuses on organizational and technical dimensions of EA. Therefore we use the terms *organizational architecture* and *IT architecture* (Fig. 1).

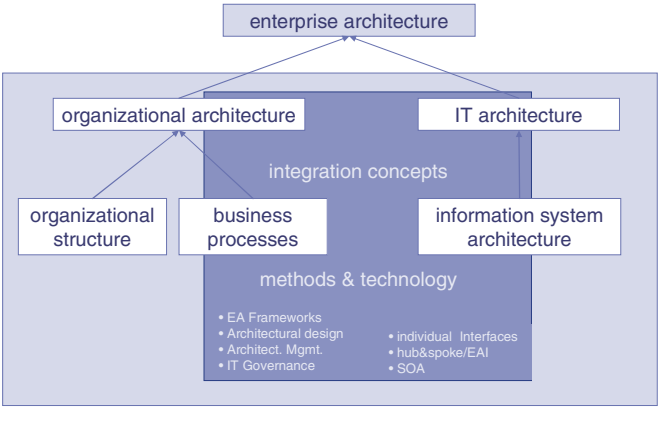

Fig. 1. Enterprise Architecture, see [1]

Organizational architecture contains all non-technical elements of the EA and is best compared with the so called instrumental understanding of organizations which covers all general explicit regulations to define the operational and organizational

structure. Accordingly we differentiate the organizational architecture in organizational structure and business processes. On a par with the organizational architecture is the IT architecture which contains all technical elements of the EA. In particular IT architecture covers the IS which are described with their own architecture: the IS architecture. Both architectures organizational and IT architecture will be considered being equivalents but observed separately to accommodate the fact that both architectures are extremely relevant for the organization's efficiency and unfortunately do have complex interdependencies to each other.

Scientific literature very often refers to the terms Organizational and IT Architecture but uses multiple term understandings. Depending on the authors scientific background the organizational architecture contains technical concepts too [4] and the IT architecture organizational aspects respectively. By definition SOA delivers not just concepts for connecting IS but reconcile IS and business processes. Both integration aspects are already considered in the technical definition of SOA which describe a business process driven IS integration. Therefore SOA could serve as a mediator between different elements of an EA.

After all, designing and deploying integration concepts require both—methodology and technology. This contribution will focus on aspects of methodology (for aspects of technology see [1]). Therefore the following section will deal with the need for and the difficulties of managing services in a SOA. Thereafter we will show how clustering algorithms can be employed to derive services domain clusters from enterprise architectures. Eventually we will discuss methodologies for applying the algorithms proposed in practice.

2 The Need for Managing Service Oriented Architectures

Issues in the field of SOA have been discussed heavily the last few years. Both scientists and practitioners emphasize the potential of SOA especially by reconciling business requirements and IT infrastructures as stated in the definition of enterprise architecture above by using integration concepts. Nevertheless there is the need for finding a stringent terminology hence common understanding used by the majority of the SOA community. Definitions range from a solely technology driven approach to a new management school approach on how to run the whole enterprise. To find a stable understanding for at least this paper the following definitions shall be analyzed:

> "[A service oriented architecture is] a set of components which can be invoked, and whose interface descriptions can be published and discovered." [5]

Gold et al. mainly consider technological aspects focusing on standardized interface descriptions. Additionally McCoy and Natis taking into account aspects of stakeholder, granularity, reuse and agility:

> "SOA is a software architecture that builds a topology of interfaces, interface implementations and interface calls. SOA is a relationship of services and service consumers, both software modules large enough to represent a complete business function. So, SOA is about reuse, encapsulation, interfaces, and ultimately, agility." [6]

In the context of existing software systems and the introduction of SOA as a new overall enterprise architecture integration paradigm, issues as management and optimization need to be addressed too:

> "SOA is the concept of service-enabling new and existing software; linking internal and external service-enabled software systems; and implementing an enterprise wide infrastructure to enable, manage, and optimize services use and interaction" [7] (see also [8, 9])

Aside from primary SOA terminology many authors have a common understanding of secondary characteristics. Summed up these are the distributed manner of SOA, the aspect of combining (orchestration of) rich software components (services), loose coupling of applications using services and the standardization of interface descriptions [8, 10, 11]. To summarize the relevant issues Lubinsky and Tyomkin highlight the business process driven integration and therefore derive the following three main aspects of a SOA [8]:

- Service descriptions
- Business Processes
- Service [Lifecycle] Management

Due to reasons of complexity especially in large organizations the solely technical view on SOA is not sufficient to successfully implement it. Methodological aspects need to be considered too. The differentiation between technical and methodological issues has been discussed quite early [12-14]. This paper contributes to the methodological aspects of SOA. Concerning service management as a term which summarizes methods to design and run SOA the following issues are relevant in the design time hence the phase when service characteristics as granularity and reuse are considered. Aside from basic service characteristics as mentioned, a service management defines a service lifecycle mainly to avoid service redundancy. Due to reasons of complexity according to business process requirements mapped to technically executable services, a methodology needs to be introduced to support early phases in the service lifecycle, the design time of a SOA. In the context of a SOA methodology terms as service management, domain engineering, governance, maturity and roadmaps can be found [15, 16]. The most generic approaches can be found in the field of domain engineering [17, 18]. The Domain Engineering introduced by the Carnegie Mellon SEI differentiates the following three activities [17]:

- Domain Analysis
- Domain Design
- Domain Implementation

In the context of service definition the domain analysis needs to be considered. Output of a domain analysis is a domain model representing relevant features used for the specific context. To derive a domain model SEI proposes to use the context analysis defining the extent of a domain, the domain modeling providing a description of the relevant issues and the architecture modeling creating the architectural artifacts [17].

Adopting the general methodology of SEI to the requirements of a SOA especially in the context of an architectural migration from existing IT architectures to SOA a domain model needs to be created. A service domain can be described as a specific modeling view that consists of modularly defined functionality which is necessary to support business processes and the underlying basic data. Within dedicated IT projects, these requirements have to be implemented. The main characteristics of domains are [19]:

- Domains encapsulate their functionality and data.
- Functionality is implemented redundancy-free, information is consistent.
- Functionality and data can be used everywhere, they can be combined to support ever new business processes.
- New projects can build upon existing assets, investments are secured.

The following chapters describe methodologies, tools and algorithms to define service domains based on the generic SEI approach of a domain analysis, clustering existing enterprise architectures hence business processes, information systems and interfaces.

3 Deriving Service Domain Clusters

For clustering enterprise architectures first of all a model of the respective architecture is needed. Minimally the model should include the following elements:

- Business processes, that means the consecutive activities (tasks) and their relationships,
- IT systems,
- the usage of IT systems along a process,
- and the interrelationships and interfaces among IT systems along a process.

Such an enterprise model can be considered a graph often called network, too. 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 a software system implementing those algorithms for enterprise architectures.

3.1 Graph Theory and Clustering Approaches

A graph consists of vertices V and edges E. All elements of our model (activities and IT systems) can be considered vertices and their relations can be considered edges. If a connection between two IT systems is used several times along a process the model will have several edges between two vertices. These edges may also be combined into a single edge with the weight w. In this case the graph will be called weighted graph.

A network with n vertices is represented by an $n \times n$ adjacency matrix A with elements

$$A_{ij} = \begin{cases} 1 & \text{if the vertices } i \text{ and } j \text{ are connected} \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

In a weighted graph A_{ij} represents the weight of an edge between the vertices *i* and *j* [20].

The partitioning of such graphs into several modules is called clustering while a cluster consists of elements that are all similar to each other in some way [21]. In our case of an enterprise architecture similarity means that vertices have a common subset of neighbors and are dependent from each other in some way. Typically this is a business activity which depends on the availability of an IT system.

Clustering approaches have a certain tradition in the analysis of social networks [22, 23]. Girvan/Newman proposed a clustering algorithm to identify communities in social networks [24]. Such a network consists of individuals (vertices) who know each other and thus have a relationship (edge). A community is defined by a number of people who know each other.

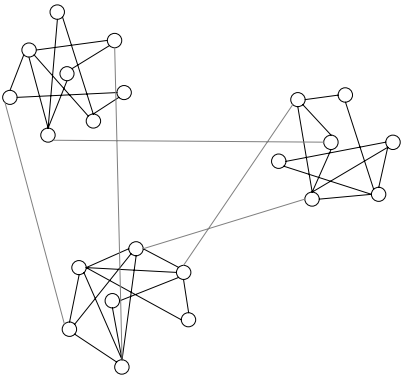

Fig. 2. A network with three communities, see [24]

Based on the analysis of shortcomings of existing clustering algorithms Girvan/Newman developed a "betweenness" algorithm. The basic idea of their algorithm is to remove the edges that are most in between in a network. The remaining vertices connected to each other form the communities. Therefore they generalized Freeman's betweenness [25] to edges and defined the edge betweenness of an edge as the number of shortest paths between pairs of vertices that run along it. The shortest paths between communities run along only a few edges. That it is why these edges will have a rather high edge betweenness. By removing these edges one can separate the communities. The algorithm valid for a weighted network is the following [20]:

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 betweennesses for all edges affected by the removal.
5. Repeat from step 3 until no edges remain.

In various papers they proved the performance of their algorithm in determining communities in social networks [20, 24, 26].

On an abstract level the problem of identifying modules in enterprise architectures is identical with the problem described for social networks. By applying the algorithm on the enterprise architecture model we identify appropriate modules as candidates for service domains in a SOA. Since one has to repeat the steps three to five for the weighted network several times it is also possible to identify a service domain hierarchy and eventually services. In every run through the algorithm, already identified modules will be further separated.

One of the important questions is when to stop the algorithm practically. Because it obviously does not make any sense to run through the algorithm as stated until no edges remain. This results in *n* modules which equals the number of vertices. So the question is when is a "good" modularity reached? Girvan/Newman propose the modularity *Q* as an indicator for the quality of clustering results [26].

Therefore they calculate the fraction of edges that fall within a module.

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

Where c_i is the module the vertex *i* belongs to. The function $\delta(u,v)$ is 1 if u = v and 0 otherwise, and $m = \frac{1}{2}\sum_{ij} A_{ij}$ is the number of edges in the graph. If the degrees *k* of vertices (the degree k_i of a vertex *i* is defined by the number of vertices connected with *i*) in a network are preserved but otherwise connect vertices together at random, then the probability of an edge existing between vertices *i* and *j* is $k_i k_j / 2m$, where k_i is the degree of vertex *i*. Thus the modularity *Q*, is given by

$$Q = \frac{1}{2m}\sum_{ij}\left[A_{ij} - \frac{k_i k_j}{2m}\right]\delta(c_i,c_j) \quad \text{[26]} \tag{3}$$

Values for Q range between 0 and 1. A value of 0 indicates a poor clustering result while 1 is a perfect cluster but a rather theoretical value. Usually values between 0.3 and 0.7 indicate good values for realistic examples.

3.2 EA Builder Software System

For modeling, analyzing and clustering we developed a software system called EA Builder [27]. The software system supports modeling of business processes, organizational structures and IT systems (Fig. 3).

Contrary to a broad range of existing enterprise class modeling solutions the metamodel of EA Builder also supports the modeling of IT systems integration on process level—regardless whether the integration solution is implemented through a middleware, SOA or classic point-to-point (P2P) interfaces.

The performance of the clustering algorithm for enterprise architecture models has been verified on a number of specially prepared test cases discussed in [28]. In the following figures we will demonstrate the functioning of the EA Builder software system with the fictitious medium-sized company WMYPC (We Make Your PC). The company sells customized computer systems starting from small multimedia

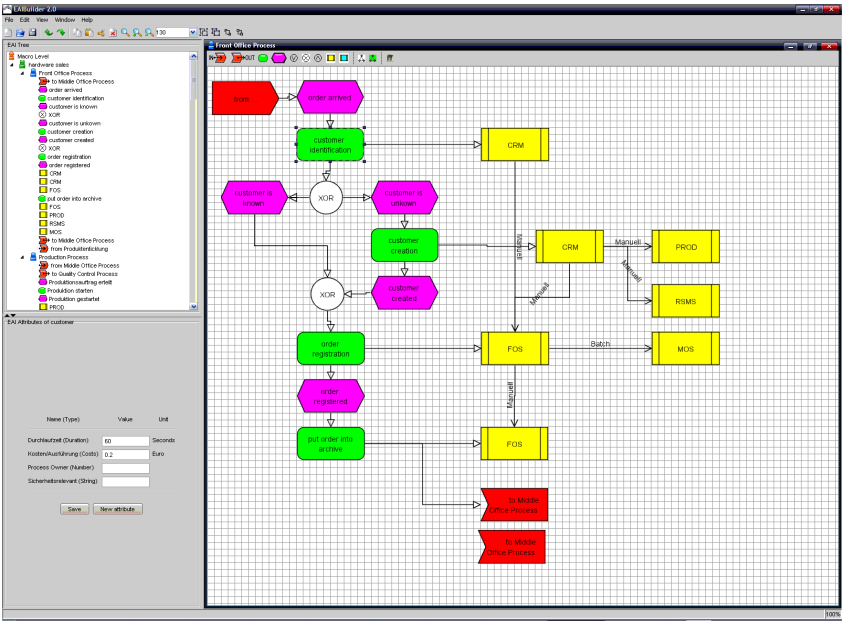

Fig. 3. EA model modeled in the EA Builder system; the screenshot shows an extended event driven process chain incorporating various IT systems which again are integrated with other IT systems along the process

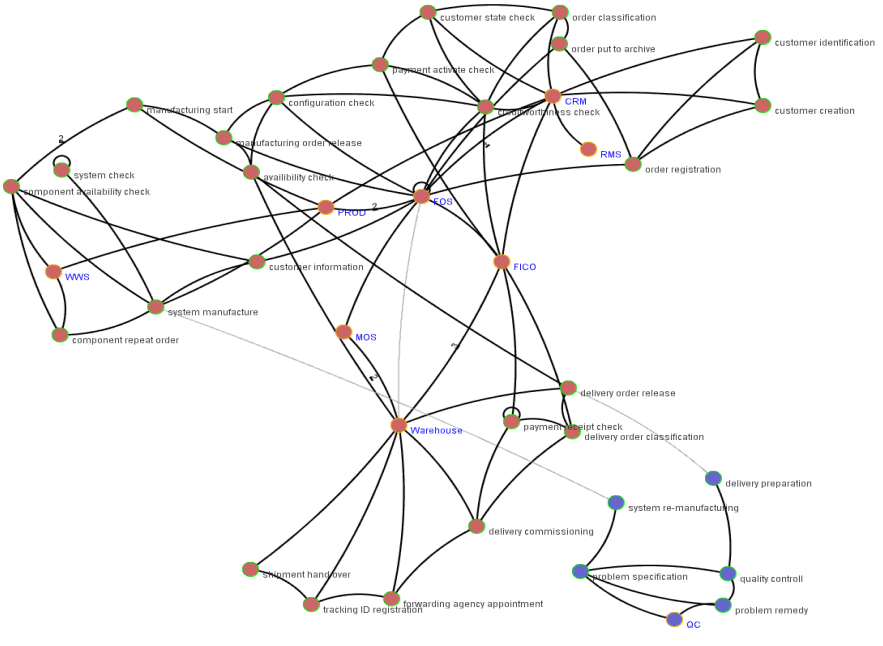

Fig. 4. Screenshot of EA Builder model transformed in a graph, 3 edges removed resulting in 2 clusters

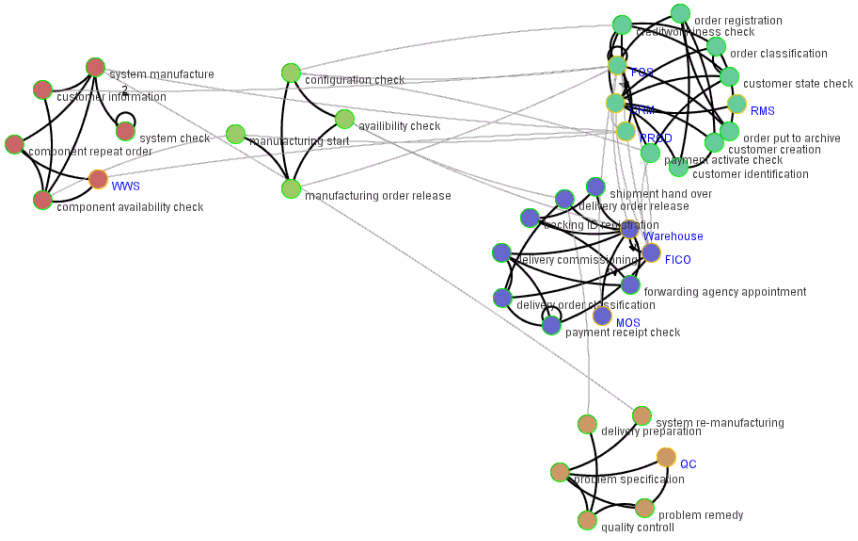

Fig. 5. Screenshot of EA Builder model transformed in a graph, 20 edges removed resulting in 5 clusters, grouped view

computers to enterprise class server systems. The processes are supported by six individually implemented information systems and one off the shelf software product. The aim of the test scenario was to realize an adequately complex business environment with business processes selectively supported by heterogeneous information systems.

Figure 4 shows an example EA model transformed into a graph. After applying the algorithm and thus removing 20 edges (figure 5) the graph is split in 5 clusters being candidates for service domains.

The resulting clusters are the first candidates for service domains. All the elements within a domain (cluster)—business processes and IT systems—have strong relationships to each other and relatively weak relationships to elements of other domains.

4 Methodology for Building Service Domain Clusters

The classic scenario for the application of our approach is the redesign of existing complex IT landscapes in a service oriented fashion. We call this bottom-up approach since we derive a service architecture from an existing complex enterprise architecture. This will be the scenario for the majority of organizations thinking about the introduction of a service oriented architecture. Compared to a top-down approach this bottom-up approach leads to a potentially smooth transition from an existing architecture to the target architecture.

A frequently stated requirement is the alignment of business processes and IT systems [29, 30]. This is exactly what we achieve through our approach. Once the EA model is transformed into a graph, we do not differentiate processes and IT systems for the application of the clustering algorithm. This means that we do not derive optimized process clusters or IT clusters—although this would be possible too. The

clusters derived reflect the actual interplay of business processes and IT systems. Thus the resulting service domain clusters consisting of business processes and IT systems are potentially well designed concerning Business-IT alignment.

However, the resulting clusters reflect the as-is structure of the modeled company "only". The as-is structure does not necessarily correspond with an existing organization chart or IT map, since it will show the actual structure—not the supposed one. Depending on the overall integration strategy it now has to be decided how to proceed with this information. One possible scenario would be to define an internal integration approach per service domain and then an additional approach for the integration of the different domains. This results in a stable architecture with local flexibility through encapsulated service domains. Such encapsulated domains may be desirable for a variety of reasons—technology, business requirements, and also politics. Eventually encapsulation leads to a better manageable complexity of enterprise architecture.

5 Conclusion

The paradigm of Service Oriented Architectures (SOA) potentially leads to more flexible enterprise architectures and an improved Business-IT alignment. Besides technological means, methodologies for the implementation of SOA in existing and complex enterprise architectures are required in order to realize the benefits promised. Building service domain models in a bottom-up approach may be a step in the right direction. In this paper we introduced a software system—the EA Builder—which derives service domain candidates based on the clustering of enterprise architecture models. A key feature of our approach is the simple but as it seems effective way of achieving a true business process oriented design by clustering business processes and IT systems in a common network.

The artificial models, we tested our approach with so far, showed some very good results. The next steps of improvement will need real world examples of grown complex enterprise architectures to show the potential of this clustering approach.

References

1. Aier, S., Schönherr, M.: Evaluating Integration Architectures – A scenariao-based Evaluation of Integration Technologies. In: Draheim, D., Weber, G. (eds.) TEAA 2005. LNCS, vol. 3888, pp. 3–16. Springer, Heidelberg (2006)
2. Aier, S., Schönherr, M.: Sustainable Enterprise Architecture with EAI – An Empirical Study. In: Milutinovic, V. (ed.) Proceedings of the International Conference on Advances in Internet, Processing, Systems, and Interdisciplinary Research, IPSI-2005, MIT Cambridge/IPSI, Cambridge, Boston MA (2005)
3. Bass, L., Clements, P., Kazman, R.: Software Architecture in Practice, 2nd edn. Pearson Education Inc., Boston (2003)
4. Nadler, D.A., Gerstein, M.S., Shaw, R.B.: Organizational Architecture – Designs for Changing Organizations. Jossey-Bass, San Francisco (1992)
5. Gold, N., Knight, C., Mohan, A., et al.: Understanding Service-Oriented Software. IEEE Software, pp. 71–77. IEEE Computer Society Press, Los Alamitos (2004)
6. McCoy, D., Natis, Y.: Service-Oriented Architecture: Mainstream Straight Ahead Gartner Research (2003)

7. New Rowley Group: Building a more flexible and efficient IT infrastructure – Moving from a conceptual SOA to a service-based infrastructure (2003) http://www.newrowley.com/reseach.html
8. Lubblinsky, B., Tyomkin, D.: Dissecting Service-Oriented Architectures. Business Integration Journal, 52–58 (2003)
9. Roth, P.: Moving to A Service Based Architecture. Business Integration Journal, 48–50 (2003)
10. Sleeper, B., Robins, B.: The Laws of Evolution: A Pragmatic Analysis of the Emerging Web Services Market. The Stencil Group, San Francisco (2002)
11. Weinreich, R., Sametinger, J.: Component Models and Component Services: Concepts and Principles. In: Council, W.T., Heinemann, G.T. (eds.) Component-Based Software Engineering: Putting Pieces Together, pp. 22–64. Addison Wesley, Boston (2001)
12. Hagel, J., Brown, J.S.: Your Next IT Strategy. Harvard Business Review 79, 105–113 (2001)
13. Gisolfi, D.: Web Services Architecture: Part 1- An Introduction to Dynamic e-business (2001), http://www-106.ibm.com/developerworks/webservices/library/ws-arcl/
14. Kirtland, M.: A Platform for Web Services. Microsoft Developer Network (2001), http://msdn.microsoft.com/library/default.asp?/en-us/dnwebsrv/html/websvcs_platform.asp
15. IBM Corporation (2005), http://www-128.ibm.com/developerworks/webservices/library/ws-soa-simm/
16. Sonic (2006), www.sonicsoftware.com/soamm
17. SEI (2004), http://www.sei.cmu.edu/domain-engineering/
18. Open Group (2006), http://www.opengroup.org/architecture/togaf8-doc/arch/p4/maturity/mat.htm
19. Bath, U., Herr, M.: 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 (2004)
20. Newman, M.E.J.: Analysis of Weighted Networks. In: Phys. Rev. E, 70 (2004)
21. O'Madadhain, J., Fisher, D., Smyth, P., et al.: Analysis and Visualization of Network Data using JUNG. Journal of Statistical Software (2005)
22. Wasserman, S., Faust, K.: Social Network Analysis: Methods and Applications. Cambridge Univ. Press, Cambridge (1999)
23. Scott, J.: Social Network Analysis: A Handbook, 2nd edn. Sage, London (2005)
24. Girvan, M., Newman, M.E.J.: Community Structure in Social and Biological Networks. Proceedings of the National Academy of Science 99, 7821–7826 (2002)
25. Freeman, L.C.: A Set of Measures of Centrality based upon Betweenness. Sociometry 40, 35–41 (1977)
26. Newman, M.E.J., Girvan, M.: Finding and Evaluating Community Structure in Networks. Phys. Rev. E 69 (2004)
27. Aier, S.: Public Information on EA Builder on the Internet (2006), http://www.ea-builder.com
28. Aier, S.: How Clustering Enterprise Architectures helps to Design Service Oriented Architectures. In: SCC'06. Proceedings of the IEEE International Conference on Services Computing, Chicago, pp. 269–272. IEEE Computer Society Press, Los Alamitos (2006)
29. Duffy, J.: IT/Business Alignment: Delivering Results (2001), http://www.cio.com/analyst/123101_idc.html
30. Luftman, J.: Measure Your Business-IT Alignment (2003), http://www.optimizemag.com/article/showArticle.jhtml?articleId=17701026