A Comprehensive Enterprise Architecture Metamodel and Its Implementation Using a Metamodeling Platform

Christian Braun, Robert Winter

Institute for Information Management
University of St. Gallen
Müller-Friedberg-Strasse 8, 9000 St. Gallen, Switzerland
Christian.Braun@unisg.ch, Robert.Winter@unisg.ch

Abstract: Due to the growing importance of alignment, compliance and manageability issues, increased attention is being paid to architectures and architecture management recently. A holistic approach to enterprise architecture requires business related and information systems related artifacts to be equally treated. This paper describes the extension of an existing approach to enterprise architecture and its implementation using a commercial metamodeling platform. In addition to the approach metamodels in general, special attention is paid to the linkages between different architecture layers, both in the underlying model and in its implementation.

1 Introduction

Information systems (IS) are instruments for the support of (informational and transactional) business processes. Hence their functionalities should be designed according to business requirements. However, significant discrepancies between IS functionalities and business requirements can be observed in real life situations: Strategic plans, organizational structures and IS often are not developed in a synchronized way, but follow different life cycles. While strategic change programs have to be effective in relatively short time frames (within a year) and while large organizational redesigns often take up to two years, the basic design of operational IS often dates back to the 1980ies or even 1970ies [Wi04].

The need for coordinating strategic positioning, organizational structures and business processes on the one hand and IS design on the other hand has been proposed by IS research for some time [Fr02]. Various approaches to enterprise architecture have been proposed to support this coordination task on various levels of abstraction (see for instance [FHG98], [Fr95], [Gr98], [HB96], [He00], [OM03], [Te99]). Often, the different levels of abstraction reflect the well known cultural differences between business people and information technology professionals [Fr02]. Holistic approaches to enterprise architecture deploy a multi-level framework or a hierarchy of design layers in order to represent the different views on an enterprise (see for instance [FS95], [Kr90], [ÖW03], [Sc98], [TO02], [ZS92]). But these multi-layer approaches are often rather abstract, do not consider all necessary design layers, or do not specify consistency in adequate rigor.

One indicator for these shortcomings is a partial, aggregate, or even completely missing metamodel that specifies consistency of the various architecture artifacts on different layers and in different views. It should also be noted that enterprise architecture is not primarily intended to (only) support IS development. As a passive instrument, it should help to identify inconsistencies between product specifications, performance indicators / goals, business process specifications, informational structures / informational flows, application design, IS functionalities, and other architecture artifacts. If used in an active way, enterprise architecture should guide redesign processes on all layers of the enterprise: strategic changes, organizational redesign as well as IS development.

In order to comply with these requirements, enterprise architecture must be constantly held current and easy to adapt. Reports and visualizations must be flexible and oriented at the requirements of the different stakeholders. An architecture management process must be in place that guarantees the professional development, alignment, and enforcement of the different architectures and artifacts. Efficient interfaces to operational development platforms must be in place that guarantee cost-efficient, effective cascading of artifact changes.

Hence the challenge of enterprise architecture is twofold: On the one hand, a metamodel is needed that is capable of representing all relevant artifacts of enterprise architecture consistently, both from a business perspective and from an IS perspective. On the other hand, such a metamodel has to be implemented using an appropriate modeling tool in order to support efficient, computer based architecture management processes, for instance the execution of consistency checks and impact analysis.

Based on a discussion of modeling issues of enterprise architecture in Section 2, the construction of an appropriate metamodel is documented in Section 3. Section 4 describes the implementation requirements for such a metamodel and the subsequent implementation using a commercial metamodeling platform.

2 Modeling Enterprise Architecture

Architecture is defined and used in many different ways (see for instance [He93], [Kr90], [Sc98], [Za87]). Most definitions have in common that architectures are aggregate representations of some complex holistic entity, i.e. they are modeling aggregate systems components and their interrelationships. In this paper, architecture is defined as any socio-technical system. It encompasses the building plan that specifies the components and their relationships as well as the design rules for developing the building plan. Architecture models support design and development processes [Kr03]. Enterprise architecture therefore should represent all aggregate artifacts that are relevant for an enterprise. The complex structure of an enterprise however can be modeled from many different views and for many different purposes [Wi03c]. A layered architecture framework is useful in order to represent different aspects in different models as long as overall consistency is preserved by appropriately modeling the frameworks' metamodel.

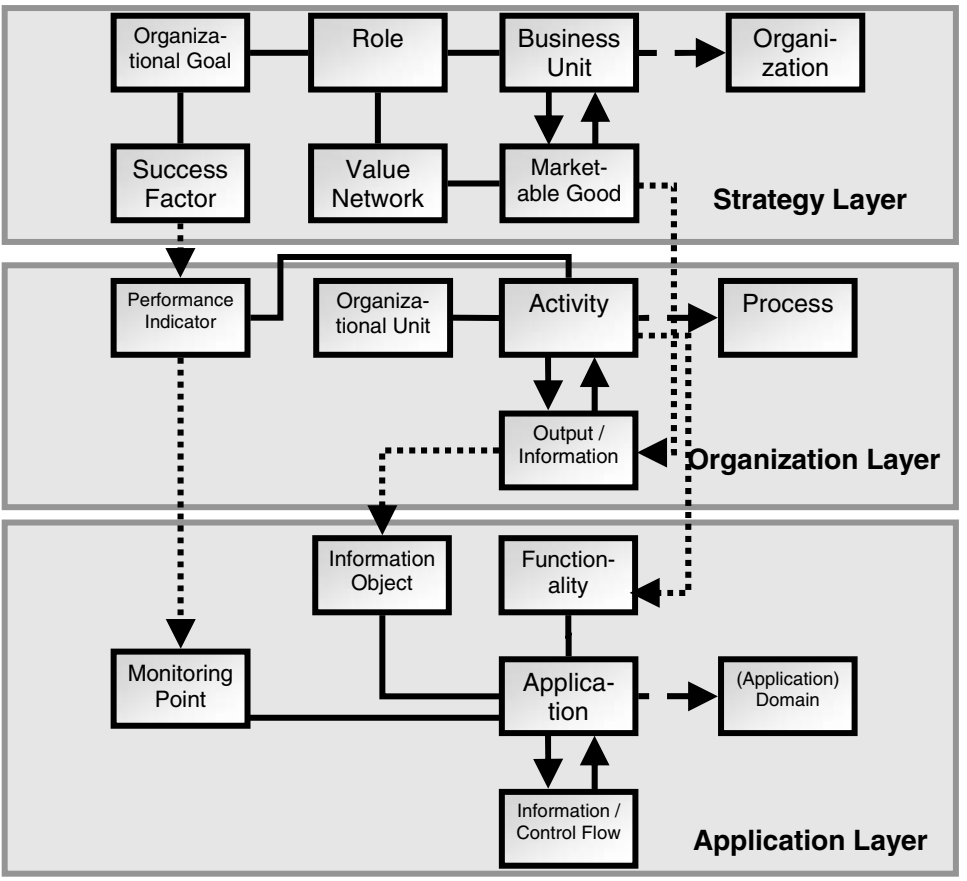

Figure 1: Aggregate Framework Metamodel (Layers 1-3)

2.1 Layered Enterprise Architecture

The analysis of related work on enterprise architecture shows that four architectural layers and thus design layers have to be differentiated [Wi03c]:

1. The general design goal on the **strategy layer** is the positioning of the enterprise, respectively the business unit, in the value network. Furthermore, the specification of product / services and the specification of organizational goals are located on this layer.

2. On the **organizational layer,** the general design goals are the efficiency and effectiveness of product / service innovation, product / service production and product / service distribution. The most important resulting specifications are business processes, key performance indicators, and organizational structure (organizational units and their relationships). In addition, information needs of business and management

processes are represented by high-level information models.

3. The design goal on the **application layer** is linking information system components to business requirements by designing appropriate application structure and by appropriate design of integration systems (e.g. data warehouse, operational data stores, enterprise application integration systems). Applications are representing high-level conceptual constructs that are used to structure information flows, process support and information systems responsibilities on a company-wide scale. Application structures are derived from analyzing activities, information usage / creation and responsibilities.

4. On the **software component layer**, the general goal is design for reuse (and design by reuse) of software artifacts and respective data structures.

Not all layers can be found in all approaches: The architecture of integrated information systems ARIS [Sc98] does not explicitly mention a strategy layer. The Zachman framework [ZS92], although including a so called 'business model', does not cover business-related artifacts like value flows, goals, organizational units and business processes in sufficient detail.

Figure 1 is a simplified version of the framework's metamodel for layers 1 through 3. In its simplified version, which is only used here for illustration purposes, regular arcs denote references between constructs. Dotted arrows denote primary dependencies between framework layers. Broken arrows denote aggregation relationships between constructs.

According to [ÖFA01], distributed structures are not only a reality for software components and information systems. An increasing number of business processes extend beyond the scope of a single business unit or even company. Even business models do more and more rely on cooperation or cooptation. The business engineering approach according to [Wi03c] intends to support the consistent specification of networked structures on all enterprise architecture layers:

On the strategy layer, value networks are specified that provide comprehensive support for certain customer processes. For an outline of modeling techniques for business network specification and business strategy specification as well as a presentation of respective notations see [Wi03b].

On the organization layer, the value network wide specifications from the strategy layer are transformed into appropriate process models that usually extend beyond the scope of a single company or business unit. Techniques for modeling business processes and organizational structures can be found in [Ös95].

On the application layer, the information and control flows between applications on different platforms, maybe in different companies, are specified. A comprehensive enterprise architecture approach has to reflect networked and distributed structures as well as ownership and management challenges induced by such structures. A model for representing application architecture is presented in [Wi03a].

2.2 Tool Support for Enterprise Modeling

Most enterprise architecture approaches are aimed at understanding relationships between different types of artifacts on an aggregate level, identifying inconsistencies between such artifacts, and providing guidelines for a consistent overall design, mostly of information systems. Driven by usability and business value considerations, enterprise architecture is more and more evolving into an active concept that supports business systems design, i.e. to contribute to the analysis and proactively support the optimization of business strategies, organizational structures, business processes, information flows, application structures as well as the underlying information systems [Sc04]. For the evolution of such an active role, it is essential that appropriate tools can be utilized: Not only is it necessary to document existing artifacts and to detect / report inconsistencies. An appropriate tool has to support the development, storage, communication / presentation and enhancement of all relevant enterprise architecture artifacts. As with enterprise architecture methodologies, enterprise architecture tools to support the architectural development process are still emerging [Sc04]. An important aspect is particularly the mapping of the dependencies between the models on different framework layers. Due to the complex nature of real-life business structures, these dependencies are often difficult to identify and understand. However it is crucial for the success of enterprise architecture management to identify such dependencies and understand the impacts of modifications among the related artifacts.

Often, it seems to be straightforward to utilize drawing tools like Microsoft Visio or Microsoft Powerpoint to document enterprise architecture artifacts. Since consistency between models and reuse of model components cannot be supported by drawing tools, serious enterprise architecture projects must use professional modeling tools from the beginning. Professional modeling tools store all artifacts in a repository and provide capabilities to reuse and present this information in different ways [Ja05]. According to the above characterization of enterprise architecture, an enterprise architecture modeling tool should be based on a repository or database that

- stores information about (a) products / services, (b) goals, success factors, performance indicators and monitoring points, (c) value networks, roles and service flows within networks, (d) activities, organizational units and control flows, (e) information objects and information flows, (f) applications, functionalities and ownership as well as (g) software components and maybe even (h) IT platforms,

- is based on a comprehensive metamodel that relates these artifacts and that

- is capable of representing this information in customizable graphical and textual forms.

The aggregate and comprehensive nature of enterprise architecture makes it necessary to complement the design of the repository with a specification of the interfaces to those systems that manage artifact specification in full detail for operational purposes (e.g.

application management systems, product data management systems, performance management systems).

3 Enterprise Architecture Based on the BAI Method

Based on the approach described in section 2.1, a couple of methods have been developed that are focusing on different business sectors or specific problem areas. One of these methods is the BAI method [Ch03] which constitutes the foundation for this paper. The BAI project developed modeling concepts for sales and distribution processes in retail banking. Based on successful practices in several analyzed banks, enterprise specific models as well as reference models for the strategy, organization and application layer were identified. In addition, the BAI method provides a procedure model for the consistent design of layers 1 through 3 of the framework and specifies the dependencies between those layers. The BAI method is structured according to the guidelines of method engineering as defined in [Gu94] and [He93]. For each framework layer, procedure models, activity specifications, modeling technique specifications, and a metamodel for all specification documents are provided. To represent the dependencies between layers, the BAI method provides an overall enterprise metamodel.

To some extent, the BAI method corresponds with MEMO [Fr02], e.g. by being based on some similar concepts. The main difference is that MEMO uses a matrix to define layers (called perspectives) and views. This simplifies the definition of relationships between elements on different layers. But it also reduces flexibility on defining different views on different layers, since the views respectively columns of the matrix are defined for all layers.

The following sections briefly summarize the metamodels of the BAI method. Additionally, requirements for a modeling technique and selected metamodels will be presented for each layer. Since this paper focuses on the metamodeling and cross-layer aspects, the method components ‘procedure model’ and ‘role’ are omitted.

For the representation of metamodels, UML class diagrams are used. These are presented in a simplified form in order to give a better overview: Cardinalities as well as some classes are suppressed if not relevant for the illustration of dependencies. The focus is on dependencies between models within a framework layer as well as between models on different framework layers.

3.1 Strategy Layer

The design of the strategy layer requires the identification of potentials of information technology and the consideration of restrictions concerning their implementation. Basic requirements for a modeling technique on the strategy layer are the modeling of companies (or business units) in their joint coverage of complex customer processes, the description of basic elements of the strategic positioning of a company (or business unit), and the definition of appropriate goals and performance indicators. Figure 2 is the meta-

model of the strategy layer in a slightly simplified form. The different parts of the metamodel that define different views are marked with different shades of gray: The strategy layer comprises the value network view, the customer process view, the service view and the performance indicator view.

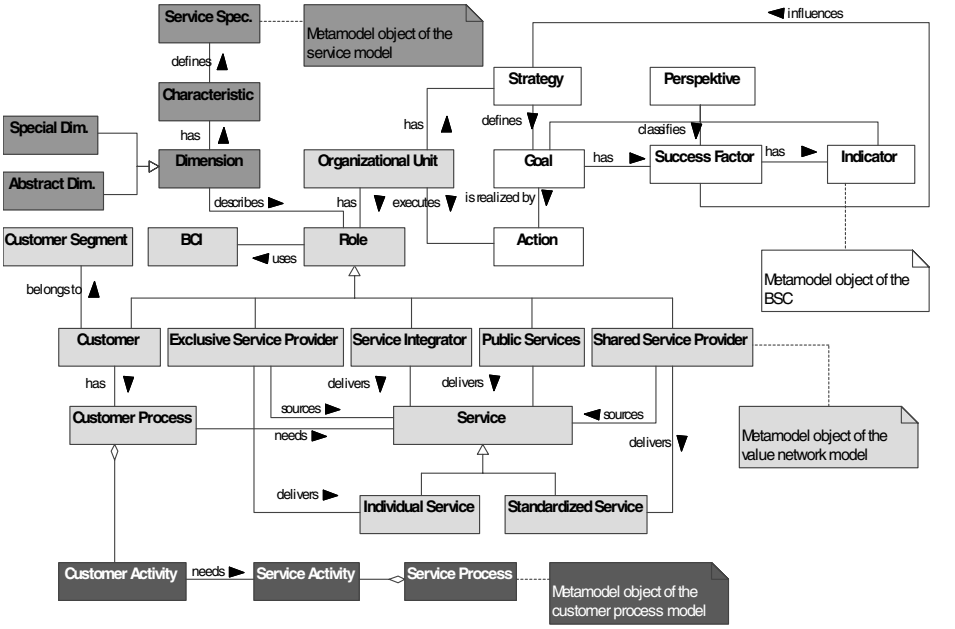

Figure 2: Strategy Layer Metamodel

The **value network view** describes the company (or business unit) according to its role(s) in the value network. Additionally, the most important service flows from and to the respective unit are specified. The **customer process view** serves as a foundation for the derivation of requirements for business process design. It structures the partial stages of the customer process and their succession. Furthermore, it defines partial services that have to be created in order to support the single partial stages within the customer process. These partial services can be created by the respective company itself or can be sourced from external service providers. The **service view** describes the strategic alignment of a company (or a business unit) regarding certain given (e.g. market) and designable (e.g. products, organization) dimensions. The **balanced scorecard view** (BSC) is used to specify performance indicators that serve as foundation for the design and performance management of business processes.

3.2 Organization Layer

The design of the organization layer constitutes the connector between the strategy and application layer. Basic requirements for a modeling technique on the process layer are a map that gives an overview about the business processes, support processes and man-

agement processes of a company (or a business unit), a clear definition of business processes by means of their control and information flows, a description of manipulated objects, applied resources and responsible roles or organizational units in a business process, and the ability to control the performance of an business process. Figure 3 is the metamodel of the organization layer, again in a slightly simplified form according to this paper's focus. Here again, the different shades of grey denote different views on the metamodel: The organization layer comprises the process map, the process control, the process control flow and the organizational structure.

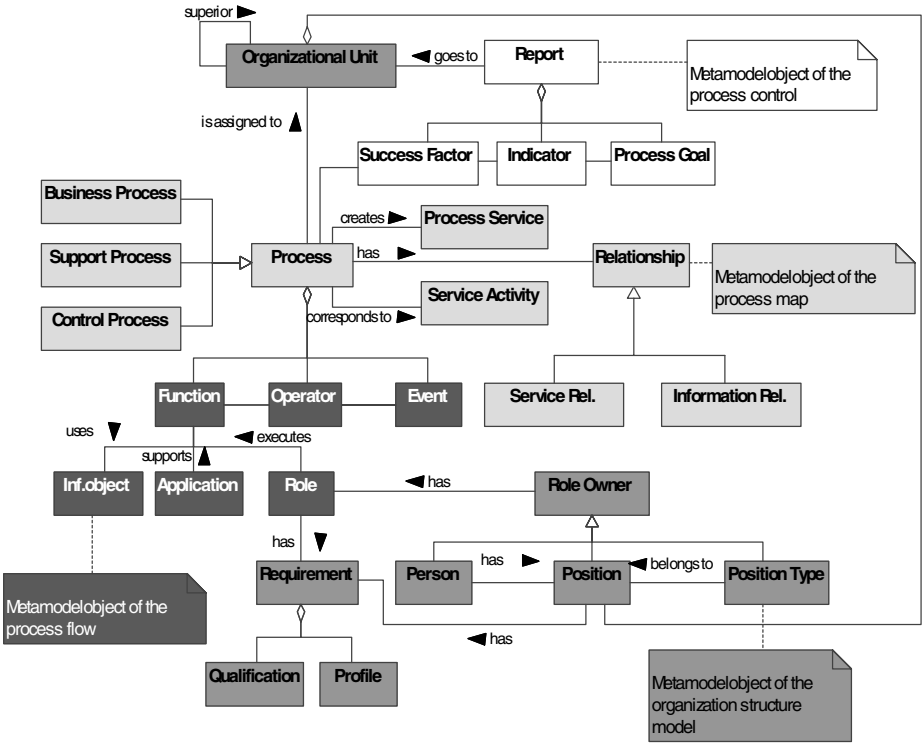

Figure 3: Organization Layer Metamodel

The **process map** is an overview about the most important business processes of the company (or business unit) in focus as well as relevant business processes of business partners in the value network; Also product / service flows between these processes are represented. The management of a certain business process is determined by the **process control**: It defines performance indicators and triggers required actions. Performance indicators are derived from performance indicators / success factors defined in the balanced score card on the strategy layer. The **process control flow** defines the elementary activities of a certain business process and their sequence as well as the assignment of ownership for activities. Additionally, it determines applications and information objects that are used for the execution of certain activities. The **organization structure** de-

scribes business units, positions, persons, roles and their (hierarchical) relationships.

3.3 Application Layer

As described above, the application layer links the business requirements for IT support as defined on the organization layer to the information systems components as specified on the software layer. The software layer is not covered by the BAI method since modularization and component specification techniques from other methods can be easily integrated. The most important development result on the application layer is an appropriate structuring and interfacing of applications. Applications are defined as logical structures that cluster certain support functionalities based on similar data access, joint ownership, joint process support or functional reuse. Figure 4 is the metamodel of the application layer, again in a slightly simplified form according to this paper's focus. The two shades of grey denote two views on the metamodel: the information objects / business functions view and the application landscape view.

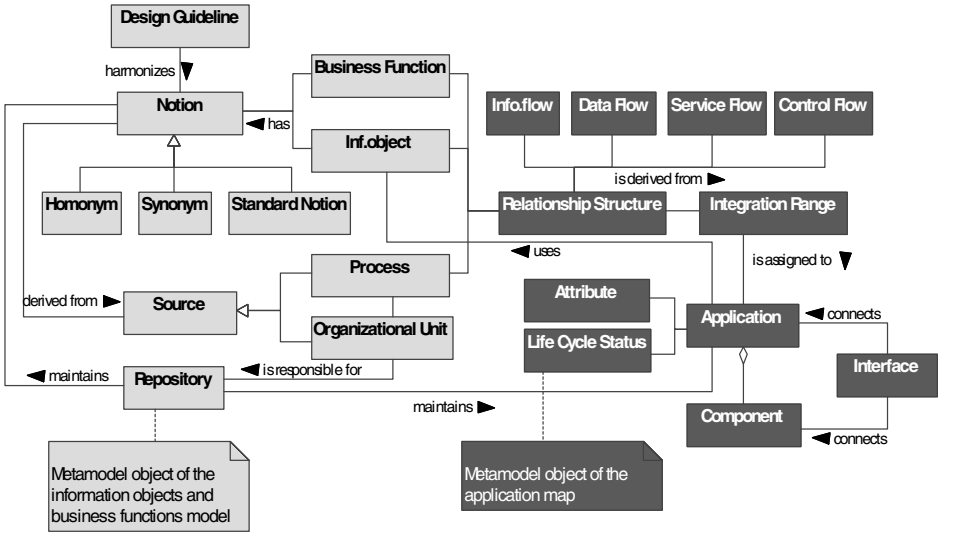

Figure 4: Application Layer Metamodel

By means of the **information objects view** and the **business functions view**, the information objects and business functions that are relevant for the business processes can be identified and documented. They have to be synchronized and standardized regarding synonyms and homonyms. They serve as foundation for the derivation of a suitable application structure. The **application landscape** specifies applications and their interdependencies. It is derived by analyzing similarities regarding data access, ownership, process support and functional reuse.

3.4 Layer Interdependencies

Figure 5 illustrates the most important interdependencies between metamodel constructs on different layers of the framework. The following numbering corresponds with the numbering used in figure 5.

(1) An organizational unit (company or business unit) holds one or more, certain role(s) within a value network. On the organization layer, business processes have to be assigned to certain organizational units to guarantee a frictionless flow and single activities have to be assigned to positions or roles that are responsible for their execution. On the application layer, organizational units are used to specify functionality ownership and hence an important source of information for identifying appropriate application structure.

(2) The characteristics of the dimensions of the product / service model are the foundations for defining outputs on the organization layer. Thus, the product / service specifications directly influence the identification of business processes.

(3) The customer process model defines which service activities have to be executed by the respective company (or business unit) and which service activities are to be sourced and integrated in order to comprehensively support a customer process. The service activities are related to the business processes defined on the organization layer.

(4) The balanced scorecard defines targets in terms of critical success factors and performance indicators which form the foundation for the design of process management on the organization layer. They are supposed to guarantee the adherence to process goals.

(5) The elementary activities that are specified as part of the process flow models on the organization layer have to be harmonized and documented in a repository regarding their different notations.

(6) Information objects that are created or manipulated by certain functionalities have also to be harmonized regarding different notations and documented in a repository. The source of an information object or a functionality can be a business process or an organizational unit. On the basis of the relationship structure between the harmonized functionalities and information objects as well as the organizational units, integration areas can be identified. These are assigned to applications which are also documented in the repository.

(7) Applications support the execution of certain activities (functionalities) of business processes.

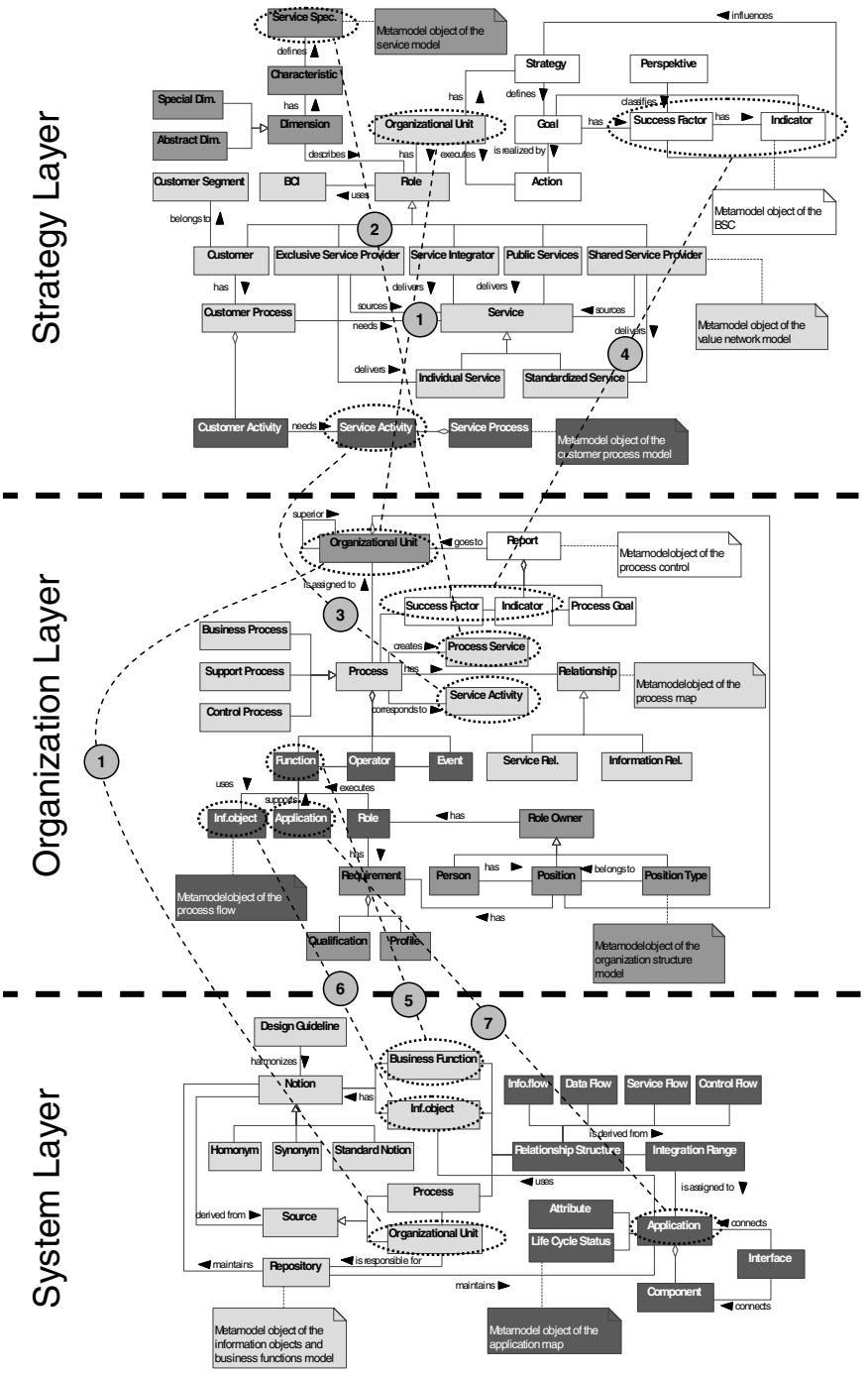

Figure 5: Layer Interdependencies

4 Implementation Using the ADONIS Metamodeling Tool

This section describes how the metamodels and their relationships can be implemented using a metamodeling tool. For this purpose, we decided to use the business process management platform ADONIS from BOC Information Technologies Consulting GmbH. The ADONIS platform is based on a method independent metamodeling approach which enables an easy customizing of the supported modeling techniques.

4.1 Special Concepts in ADONIS

In ADONIS, every method is based on an implementation library. This library can be created and customized using the ADONIS Administration Toolkit. It comprises all information for the individual, customized use of ADONIS. The library is always separated into (1) a business process library comprising information about process models and (2) a work environment library comprising information about organizational structures (e.g. organizational charts). Both comprise the definitions for model types, classes and relationship classes. A model type is a grouping of classes. Classes represent the template for objects created by the modeler. Classes have attributes which determine for example the graphical representation of an object or the arrangement of object attributes. Furthermore, the classes also define object attributes. Every object attribute has an attribute type and a standard value. In the modeling component of ADONIS, models (e.g. business process models) are created based on a specific model type. Models own attributes which contain general information about the model (e.g. creation date, status). A model comprises objects which are derived (instantiated) from classes. Objects own attributes which comprise the information for the description of the model content.

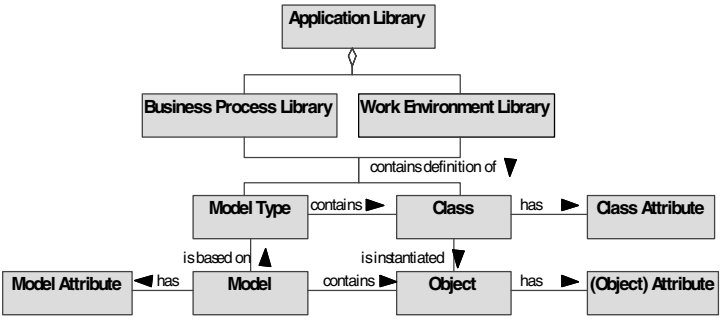

Figure 6: Concepts and relationships used in ADONIS

4.2 Mapping of Metamodels

The mapping of the metamodels described in section 3 necessitates the creation and customization of a new implementation library. All metamodels of the strategy, organization and application layer are mapped to the business process library. Merely the metamodel of the organizational structure is mapped to the work environment library.

First of all, for every object type of the metamodel a new instantiable class has to be created in ADONIS. Every class owns attributes which define for example the graphical representation on the user interface, the alignment of attributes in the ADONIS notebooks and the links to other models. For every relationship type that connects two object types of a metamodel, for example 'sources service', a relationship class can be defined in ADONIS. Dependencies or relationships between objects in different models can be represented via a class attribute typed 'reference'. For example, an instance of the object type 'service integrator' in the value network model has always a reference to the according instance of the object type 'organizational unit'. The definition of a reference enables the navigation from an object to an object in another model. After defining all classes as well as their attributes, a corresponding model type in ADONIS has to be defined for every metamodel. A model type determines a subset of all instantiable classes and relationships. Any model that is created with the model editor has a certain model type which can not be modified afterwards.

4.3 Description of the Prototype

The semi-formal specification of the metamodels in UML class diagrams allowed for the generation of a software prototype. A first version of this prototype is already available and will be briefly described by means of a case example in this section. Generally, all metamodels described in section 3 were mapped to model types in ADONIS. When creating a new model, a dialog shows all model types grouped according to the strategy, organization and application layer. After choosing a certain model type, the model editor is opened. On the left side of the model editor are listed all classes that are available for this model type. By choosing a class, objects of this class can be placed on the workspace. As an application example, Figure 7 illustrates the results of modeling a value network, a customer process, an organization structure, a process map, a process flow and an application repository. Since the focus of this paper is on the dependencies between models on different framework layers, the implementation of some dependencies will be briefly described.

A role within the value network model is always associated with an organizational unit (company or business unit). By clicking on a role object, the modeler can directly navigate to the according organizational unit within the organization chart. Another reference exists between a certain customer process within the value network model and its detailed representation with all partial activities and services within the according customer process model. By clicking on the customer process object in the value network model, the customer process model is opened in the workspace. The services produced on the process layer are maintained hierarchically in a pool model. They can be used for modeling a process map and a detailed process flow. Furthermore, for every service, one or more references on a service specification can be defined. A process within a process map can either reference on a more detailed process map or on the according process flow, illustrated as eEPC (extended Event Process Chain). Within a process flow, a function can reference on a sub-process. Thus, a hierarchically process modeling with different levels of detail is supported. Critical success factors and performance indicators of the process control correspond to the critical success factors and performance indicators

defined in the balanced scorecard on the strategy layer. This correspondence is also realized by an object reference between different models. Additionally, performance indicators on the process layer can be illustrated as references on the organization layer within the process control. The functions and information objects on the organization layer possess each a reference on the according function respectively the according information object within the business functions model or information objects model. The applications derived from the harmonized business functions, information objects and organizational units are maintained in an application repository. There, they can be referenced for modeling the process flows.

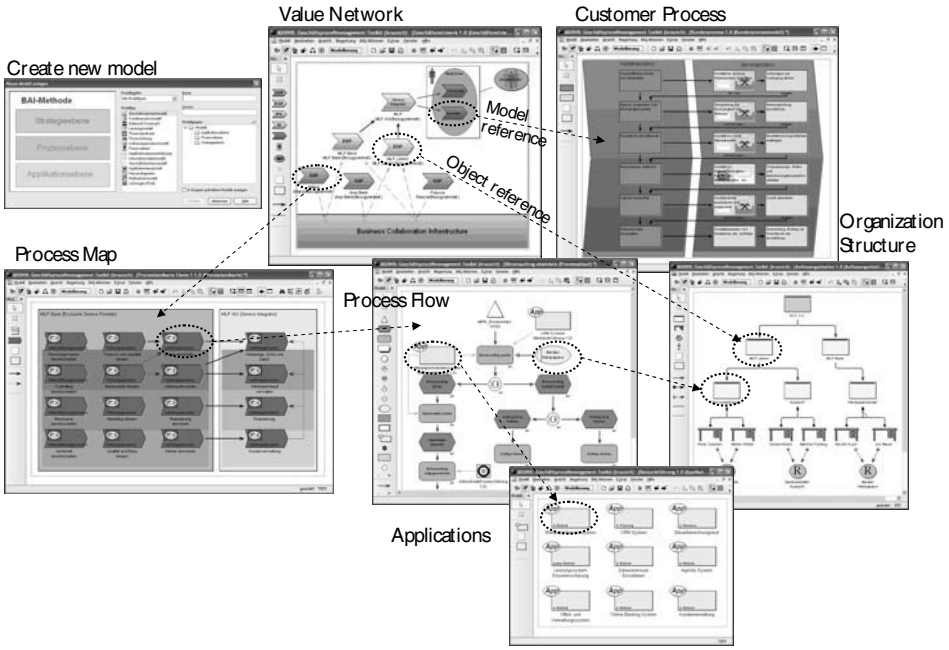

Figure 7: Model and Object References

5 Conclusions and Future Work

In this paper we discussed the intra- and inter-layer consistency issues associated with a comprehensive enterprise architecture approach that is aimed at covering business related artifacts and information systems related artifacts in comparable detail. The applicability of the proposed approach and its metamodel was shown by means of a modeling tool prototype based on the ADONIS metamodeling platform.

Since this paper focuses on consistency issues in multi-layered architectures, both with regard to the underlying metamodel as well as its implementation, the metamodels have not been described in detail. An upcoming dissertation will provide the detailed descriptions of all metamodel elements.

A first version of the enterprise architecture tool prototype is currently being pilot-tested in two companies. First application and evaluation results will be available by the end of 2005 and will also be published in the above mentioned dissertation. Based on these results, a beta version of the tool will be released, and a second round of pilot tests will be started in early 2006.

Of particular interest in the pilot tests are the ability of the prototype to support manageable architecture modeling (i.e. the co-existence of operational artifact management in full detail and aggregate enterprise architecture management), the degree of IT-business alignment that can be achieved by adhering to the consistency constraints implicated by the approach's metamodel, and the validation of the scope of artifacts that is covered by the current version of the repository.

References

[Ch03] Choinowski, S., Leist, S., Winter, R., Zellner, G.: BAI Methode, Working Paper, Institut für Wirtschaftsinformatik, Universität St. Gallen, St. Gallen 2003.

[FHG98] Firesmith, D., Henderson-Sellers, B., Graham, I.: Open Modeling Language Reference Manual, SIGS Books, 1998.

[Fr95] Frese, E.: Grundlagen der Organisation. Konzept - Prinzipien - Strukturen, 6. Aufl., Gabler, Wiesbaden 1995.

[Fr02] Frank, U.: Multi-Perspective Enterprise Modeling (MEMO) - Conceptual Framework and Modeling Languages, Proceedings of the Hawaii International Conference on System Sciences (HICSS-35), 2002, S. 3021.

[FS95] Ferstl, O. K., Sinz, E. J.: Der Ansatz des Semantsichen Objektmodells (SOM) zur Modellierung von Geschäftsprozessen, in: Wirtschaftsinformatik, 37. Jg., Nr. 3, 1995, S. 209-220.

[Gr98] Grant, R. M.: Contemporary strategy analysis: Concepts, techniques, applications. A guide for instructors, Blackwell Publishers, 1998.

[Gu94] Gutzwiller, T.: Das CC RIM-Referenzmodell für den Entwurf von betrieblichen, transaktionsorientierten Informationssystemen, Physica, Heidelberg 1994.

[HB96] Hess, T., Brecht, L.: State of the art des Business process redesign: Darstellung und Vergleich bestehender Methoden, 2. Aufl., Gabler, Wiesbaden 1996.

[He93] Heinrich, L. J.: Wirtschaftsinformatik: Einführung und Grundlegung, Oldenbourg, München/Wien 1993.

[He93] Heym, M.: Methoden-Engineering: Spezifikation und Integration von Entwicklungsmethoden für Informationssysteme, Rosch-Buch, Hallstadt 1993.

[He00] Heinrich, B.: Dimensionen zur Beschreibung eines Geschäftsmodells für Kreditinstitute im Bereich Privatkunden, Institut für Wirtschaftsinformatik, Universität St. Gallen, St. Gallen 2000.

[Ja05] James, G.: Magic Quadrant for Enterprise Architecture Tools, www.gartner.com (Zugriff: 19.04.2005).

[Kr90] Krcmar, H.: Bedeutung und Ziele von Informationssystemarchitekturen, in: Wirtschaftsinformatik, 32. Jg., Nr. 5, 1990, S. 395-402.

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

[ÖFA01] Österle, H.; Fleisch, E.; Alt, R.: Business Networking: Shaping Collaboration Between Enterprises, Springer, Berlin 2001.

[OM03] OMG: Model Driven Architecture (MDA) FAQ, http://www.omg.org/mda/faq_mda.htm (Zugriff: 04.05.2005).

[Ös95] Österle, H.: Business in the Information Age - Heading for New Processes, Springer, New York 1995.
[ÖW03] Österle, H., Winter, R.: Business Engineering, in: Österle, H., Winter, R. (Hrsg.): Business Engineering - Auf dem Weg zum Unternehmen des Informationszeitalters, Springer, Berlin et al.. 2003, S. 3-20.
[Sc98] Scheer, A.-W.: ARIS – Vom Geschäftsprozess zum Anwendungssystem, Springer, Berlin et al. 1998.
[Sc04] Schekkerman, J.: How to Survive in the Jungle of Enterprise Architecture Frameworks: Creating or Choosing an Enterprise Architecture Framework, Trafford Publishing, Victoria, British Columbia 2004.
[Te99] Teubner, R. A.: Organisations- und Informationssystemgestaltung, Theoretische Grundlagen und integrierte Methoden, Wiesbaden 1999.
[TO02] TOGAF: The Open Group Architecture Framework "Enterprise Edition" Version 8.1, http://www.opengroup.org/architecture/togaf8-doc/arch/ (Zugriff: 05.06.2005).
[Wi03a] Winter, R.: An Architecture Model for Supporting Application Integration Decisions, Proceedings of the 11th European Conference on Information Systems, 2003.
[Wi03b] Winter, R.: Conceptual Modeling of Business Networks and Business Strategies, eTransformation, 16th Bled eCommerce Conference, Moderna Organizacija, Kranj 2003, S. 551-568.
[Wi03c] Winter, R.: Modelle, Techniken und Werkzeuge im Business Engineering, in: Österle, H., Winter, R. (Hrsg.): Business Engineering - Auf dem Weg zum Unternehmen des Informationszeitalters, Springer, Berlin et al. 2003, S. 87-118.
[Wi04] Winter, R.: Architektur braucht Management, in: Wirtschaftsinformatik, 46. Jg., Nr. 2004, S. 317-319.
[Za87] Zachman, J. A.: A Framework for Information Systems Architecture, in: IBM Systems Journal, 26. Jg., Nr. 3, 1987, S. 276-292.
[ZS92] Zachman, J. A., Sowa, J. F.: Extending and formalizing the framework for information systems architecture, in: IBM Systems Journal, 31. Jg., Nr. 3, 1992, S. 590-616.