FAKULTÄT FÜR INFORMATIK
DER TECHNISCHEN UNIVERSITÄT MÜNCHEN

Master's Thesis in Wirtschaftsinformatik

Open Source Tools for Enterprise Architecture Management – Design and Application of an Evaluation Guideline

Open Source Tools für das Enterprise Architecture Management – Entwicklung und Anwendung eines Bewertungsleitfadens

Katharina Theresa Winter

Supervisor:	Prof. Dr. rer. nat. Florian Matthes
Advisor:	Dipl.-Inf. Sabine Buckl
Submission Date:	15.12.2010

I assure the single handed composition of this master's thesis only supported by declared resources.

Garching, 15.12.2010

.....

Abstract

In recent years, a plurality of enterprise architecture (EA) management tools has been developed. Companies planning to introduce tool support for their EA management function are forced to conduct laborious evaluations of the various tools. Driven by the demand from industry, the sebis chair of the Technische Universität München carried out two extensive surveys on EA management tools and summarized the results in the Enterprise Architecture Management Tool Survey 2005 and 2008. When looking at the ever changing market of EA management tools, the majority of tools can be described as proprietary. Recently, however, a few open source EA management tools have been contributed by initiators with different backgrounds. In the course of this thesis, the open source EA management tool Essential Project is evaluated in detail by applying the scenario-based approach taken by the Enterprise Architecture Management Tool Survey. The results from this evaluation are subsequently used to compare Essential Project with two other open source EA management tools – iteraplan and Tricia/SyCaTool. Complementing the topic of evaluating EA management tools, an enterprise-specific and scenario-based evaluation guide is proposed. Tool evaluation processes in the context of EA management tools have not yet been considered in scientific literature. Therefore, this thesis sketches an evaluation process, which was developed based on evaluation processes for software products in general and an EA management tool evaluation process conducted in practice.

Table of contents

Table of figures

1 Introduction and overview

Nowadays, application landscapes of modern enterprises comprise hundreds or even several thousand business applications. These landscapes represent intransparent and hardly governable structures, which have grown historically. Thus, changes to the application landscapes are both costly and time-consuming, but inevitable as enterprises are forced to continuously align their business and information technology (IT) to account for the dynamics in an enterprise's environment. Enterprise architecture (EA) management is therefore not only concerned with technical aspects such as business applications, interfaces, and infrastructure components, but also with concepts like business processes and strategic goals. As the few concepts mentioned previously cover only a tiny part of an enterprise architecture (EA) and exhibit close interdependencies, it becomes obvious that EA management is a complex task. It has to deal with a considerable amount of information that has to be stored, kept up-to-date, visualized, and analyzed. In addition, these tasks are often performed by distributed teams composed of different roles with different responsibilities and access rights. Taking all these facts into account, it is apparent that tool support is crucial for EA management. Over the years a multitude of EA management tools has been developed incorporating different approaches. Driven by the demand from industry partners, the sebis chair of the Technische Universität München conducted evaluations of prominent EA management tools and published the results in the *Enterprise Architecture Management Tool Survey* (EAMTS) 2005 [se05] and 2008 [Ma08]. A scenario-based approach to evaluating the tools was taken focusing on specific tool functionalities and their support for EA management. The *Enterprise Architecture Management Tool Surveys* yielded more than a simple raking of the investigated tools, instead they characterized them according to different aspects showing the specific strengths and weaknesses of each tool. Thus, enterprises can deduce on a high level and with low effort, which of the tools are appropriate for their intended usage or not. However, a decision for an EA management tool must be carefully taken as a lot of time and effort is required for introducing the selected tool. The methodology proposed by the tool must be implemented within the organization and processes of the company's EA management function and the employees need introductory training. After using the EA management tool for a while, the data captured within the tool constitutes a valuable asset to the company as a lot of time and effort has been spent on gathering and maintaining EA information. Due to the fact that different EA management tools may contain fairly different information models, replacing the current tool with a different EA management tool is difficult and rather costly. Therefore, a kind of lock-in situation emerges on top of the inevitable licensing costs of proprietary EA management tools. For this reason, open source may constitute a beneficial alternative for many companies.

Over a decade ago, a new philosophy called *open source* emerged exhibiting a different approach to using, modifying, and redistributing software. The key idea behind this new philosophy is that the source code should be available to the general public with relaxed or non-existent copyright restrictions. Open source software projects have developed and are still developing open source software products, which are well-known and widely-used such as the Apache HTTP Server and the internet browser Mozilla Firefox. In recent years, a few open source tools for EA management were proposed, but have not yet been considered in the EAMTS 2008. Therefore, the EAMTS 2008 is

complemented with evaluation results obtained from evaluating the open source EA management tool *Essential Project* in the course of this thesis.

A further point which has been neglected so far in literature is the organization-specificity of an EA management tool evaluation and selection. The process of EA management is set up quite differently in different enterprises as companies pursue different goals with their EA management approach. Therefore, the configuration of EA management and also its placement in a company's organizational and process structure varies widely. Those enterprise-specific characteristics have to be taken into account during evaluating and deciding on an EA management tool in order to ensure that a company's requirements are adequately covered. Guidelines or processes for EA management tool evaluation and selection have not yet been examined by scientific literature although they exist for general software product evaluation and selection.

1.1 Objectives of the thesis

The first objective of the thesis is to complement the EAMTS 2008 with the evaluation of *Essential Project* – an open source tool for EA management. The *Essential Project* is evaluated using the scenario-based approach employed in the EAMTS 2008 [Ma08]. The results of the evaluation are taken as a basis for comparing the *Essential Project* to the other open source EA management tools of *Iteraplan* and *SyCaTool*.

A further objective of the thesis is to develop guidelines for enterprise-specific EA management tool evaluation and selection, which are based on the EAMTS 2008. Processes for EA management tool selection have neither been described in scientific literature nor have guidelines or a generic process been proposed. Such guidelines and generic processes, however, exist for general software evaluation and selection. Based on these and on information from an interview with a company that has already conducted such an evaluation and selection process, guidelines for enterprise-specific EA management tool evaluation and selection are compiled.

1.2 Environment of the thesis

The main part of the thesis concerning the evaluation of the open source tool *Essential Project* was supported by members of the Essential Team, who characterized their EA management tool according to the questions in the list of criteria and answered any questions occurring during the evaluation of Essential Project.

Due to the practical relevance of the topic of EA management tools, there is a close cooperation of the sebis chair with industry partners in this area of research. Hence, input for the compilation of guidelines for the enterprise-specific tool evaluation and selection was provided by an employee of a German communication service provider. In this company an EA management tool evaluation and selection process based on the EAMTS 2008 has been conducted. In order to profit from the experiences made and to incorporate them in the guidelines an interview was held with the employee leading the evaluation and selection process.

1.3 Course and structure of the thesis

The subsequent Chapter 2 serves as an introduction to the topic of EA management, the EA management tool market, and open source. In the first section basic terms, such as architecture and enterprise architecture are defined, followed by a description of the typical layers of an EA. Furthermore, EA management is defined and characterized with respect to aspects like goals, methods, and popular frameworks. Section 2.2 is concerned with describing the EAMTS 2008 focusing on the evaluation approach, which constitutes the foundation for the evaluation of *Essential Project* in Chapter 3 of the thesis. In the third section the topic of open source is introduced, starting with defining open source, describing the roles and governance structures in open source software development projects and concludes with introducing in short two well-known open source software products in the context of EA management.

Chapter 3 describes the evaluation of the open source EA management tool *Essential Project*. In Section 3.1 the Essential Project is introduced by depicting its history, the components, and the tool architecture. Furthermore, the *Essential Meta-Model*, which constitutes the information model in the terminology of this thesis is delineated in detail. The adaptations made to the list of criteria given by the EAMTS 2008 [Ma08] are described in Section 3.2. This adapted list of criteria was sent to the Essential Project team and answered by them in detail. Their responses to the various questions served as an input to the actual evaluation of Essential Project, which is presented in Section 3.3. As the evaluation is based on the evaluation approach used in the EAMTS 2008, Essential Project is assessed according to the scenarios for analyzing specific functionality and the scenarios for analyzing EA management support. The concluding Section 3.4 of the evaluation of Essential Project gives an executive summary with the most important findings of the evaluation.

In Chapter 4 the three EA management tools of *Essential Project*, *Iteraplan*, and *SyCaTool* tool are compared to each other. The first two sections of this chapter describe according to the EAMTS 2008 scenarios the capabilities of the open source tools. The subsequent Section 4.3 compares the three tools and details on their specific strengths and weaknesses.

Chapter 5 is concerned with the development of an enterprise-specific evaluation guide for EA management tools. Thereby, at first processes found in scientific literature for evaluating software products in general are delineated. Based on an interview with a German communication service provider, a description of an EA management tool evaluation process in practice is provided in Section 0. Taking into account the results of the previous two sections, Section 5.3 is concerned with compiling the enterprise-specific evaluation guide for EA management tools. The concluding Chapter 6 summarizes the thesis and identifies future research potential.

2 Enterprise architecture management and tools

This chapter starts with a detailed depiction of EA management and related concepts establishing a common terminology for the remainder of the thesis. In the second section the EAMTS 2008 is delineated in detail as it is the basis for the evaluation of Essential Project in Chapter 3. The last section is concerned with introducing open source and showing its influence on EA management tools.

2.1 Enterprise architecture management

In EA management literature a variety of definitions for EA management exists. On that account the notions of architecture, EA, as well as EA management are defined and explained subsequently in order to provide a common terminology for the remainder of the thesis.

When it comes to defining EA most papers cite the ISO/IEC 42010 or ANSI/IEEE Std 1471-2000. Accordingly, architecture is defined as "the fundamental organization of a system embodied in its components, their relationships to each other, and to the environment, and the principles guiding its design and evolution" [IE00]. Based on this general definition of architecture, EA may then be defined as the coherent and holistic architecture of an enterprise, which comprises both IT and business elements. It does not solely consist of elements such as the organizational structure, business processes, applications, and infrastructure elements, but also contains their relationships and crosscutting functions like strategies & objectives, requirements & projects, blueprints & patterns, and KPIs & metrics [Wi07]. An EA description can be used in fairly different ways. On the one hand it might serve as a means for satisfying information needs of the various stakeholders while on the other hand it can also be used to perform analyses. In [ARW08] a number of usage scenarios for EA descriptions are depicted, e.g. IT/business alignment, business continuity management, and compliance management etc.

When it comes to defining EA management again a multitude of different definitions can be found in scientific literature [FAW07, La05, RWR06] and so far no common definition exists. The definition of EA management used in this thesis is the following holistic definition from the EAMTS 2008 [Ma08]:

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

Due to the goal of continually aligning business and IT, EA management is not only concerned with technical aspects, such as infrastructure components, but also with

aspects from the business side like organizational structures and business processes. Figure 1 shows the different layers that can be used to organize the various concepts relevant to EA management. The *business capability* abstraction layer describes the organization from a customer perspective, i.e. it summarizes the products and services offered as well as the business interactions. The architecture layer *business & organization* focuses on the organization- and process-related aspects of an enterprise. The organization's internal services needed for supporting and executing business processes are subsumed in the *business service* abstraction layer. The *application & information* architecture layer is concerned with the business applications and the information exchange between them. The technical services needed for supporting and operating business applications are described in the *infrastructure service* abstraction layer. Finally, the bottom *infrastructure & data* architecture layer centers around the technical basis used by the applications, i.e. the basic technologies, operating systems, and hardware devices needed to run the business applications. In addition to the architecture and abstraction layers, orthogonal cross-cutting aspects exist, which may have an influence on any of the elements organized in the layers. Linkages of cross-cutting aspects to elements in the layers may be accomplished in different ways. Firstly, a linkage may be given by measuring goals via KPIs and defined measures in EA concepts. Secondly, a linkage can indicate that projects target certain EA concepts and thirdly, a linkage may represent that specific EA concepts are subject to defined standards[1].

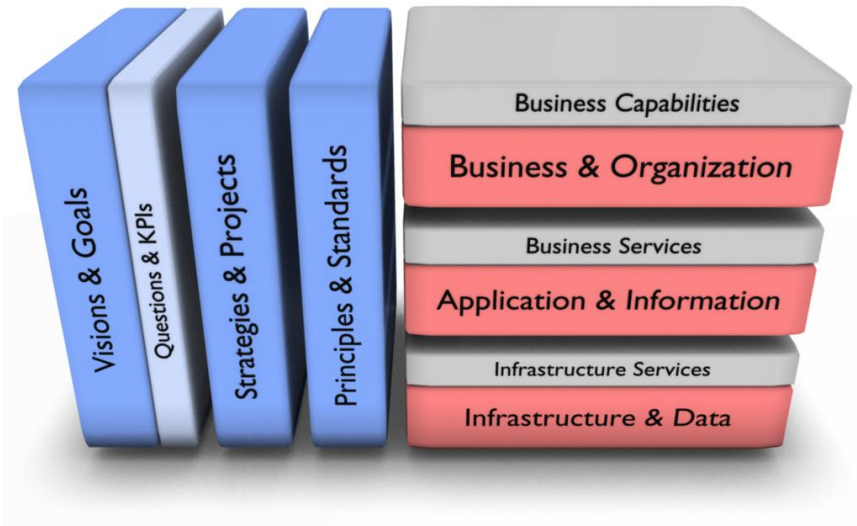


Figure 1: Architectural layers, abstraction layers and cross-cutting aspects. Source: http://wwwmatthes.in.tum.de/wikis/beams/ibb, accessed 21.10.10

[1] http://wwwmatthes.in.tum.de/wikis/beams/ibb, accessed 21.10.2010

The sebis approach with its layers and cross-cutting functions underlies the EA management perception of the EAMTS 2008. Nevertheless, the two other prominent conceptions of [FW06] and [Ni05] are introduced. As the sebis approach described in the previous paragraph shows, most EA management approaches distinguish several EA layers reaching down from business to IT infrastructure. The number of layers varies as distinct approaches choose different segmentations of an EA. [FW06] for example propose five layers starting with the *business architecture* that represents the fundamental organization of the company or governance agency from a business strategy perspective. The next layer is called *process architecture*, which stands for the fundamental organization of service development, service creation, and service distribution in an enterprise. A further layer is the *integration architecture* representing the fundamental organization of information system components in the relevant enterprise context. The fundamental organization of software artifacts, such as software services and data structures are combined in a layer named *software architecture*. The fifth layer is called *technology* or *infrastructure architecture* and represents the fundamental organization of computing and telecommunications hardware, as well as networks. [Ni05] in contrast only suggests the three layers of business, application, and systems architecture. Thereby, a *business architecture* is understood as a collection of artifacts describing the business, such as strategies, goals, organizational structure, and process structure. The *application architecture* specifies the application systems of a company, their inner structure, their technical components, and the principles according to which the systems were developed. The *systems architecture* as a third layer represents concepts concerning the company's infrastructure and system operation.

Over the last years a multitude of EA management approaches have been developed both by scientists [Fr02, HW08, La05, Ro06, We03] and practitioners [De06, Ke06, Ni05, OG09, Sc08, Za87]. The public sector has also contributed via different architecture frameworks such as DoDAF [DD09] and FEAF [CIO99]. The method prescriptions contained in some of these approaches were the basis for identifying the four EA management activities in [BMS10]. *Developing & describing* is concerned with creating descriptions of the current, planned, and target states of the EA. Thereby the current state of the EA describes the status-quo, the planned state describes a medium-term future state, and the target state stands for a description of the long-term vision. For these descriptions all levels of architectures ranging from the business & organization level, via the application & information level, to the infrastructure & data level are taken into account. Additional information is documented for projects & architectural principles and standards. Moreover, the architecture principles are established guiding the evolution from the current to the target state. The second EA management activity *communicate & enact* is concerned with spreading information on the described states of the EA and architectural principles to the enterprise-level management functions [BMS10]. In doing so, different ways of implementing the activity of communicate & enact exist, ranging from simply informing decision makers to the powerful right of stopping projects, which do not conform to the EA. As usually different states of the EA (current, planned, target) and also different scenarios of a state may exist, the *analyze & evaluate* activity makes those different states comparable and thus provides a basis for subsequent decision making. In literature, many different ways of analyzing the EA have been proposed, varying widely in their degree of formalization, ranging from expert-based assessments to indicator based computations. *Configure & adapt* deals on the one hand with setting up EA management and on the

other hand with measuring the performance of the EA management function [BMS10]. When setting up EA management it is important to define the goals and objectives of the EA management initiative. A further important setup task is to identify relevant stakeholders and their concerns. Additionally, decisions have to be made on the scope and reach of the EA management function. Once the EA management initiative has been set up, the configure & adapt activity is concerned with measuring the performance of and adapting the EA management function if necessary [BMS10].

As illustrated in the previous paragraph, EA management is made up of a multitude of tasks that are often performed by distributed teams. Moreover, an extensive amount of data on current, planned, and target states of the EA has to be stored, kept-up-to-date, consolidated, visualized, and analyzed. Therefore, the use of an EA management tool is essential for an EA management initiative. For this reason a multitude of EA management tools have been developed over the years exhibiting different strengths and weaknesses. They have been evaluated by consultants such as Gartner and Forrester Research. However, these evaluations were performed on a high level view [Bu08a] and thus leaving the evaluation of specific tool functionalities up to the companies seeking to establish EA management tool support for their EA management initiative. In order to remedy the deficiencies of those evaluations, the sebis chair of the Technische Universität München conducted scenario-based evaluations of prominent EA management tools and published the results in form of the EAMTS 2005[se05] and the EAMTS 2008 [Ma08]. The EAMTS 2008 is delineated in detail in the subsequent section.

2.2 Enterprise Architecture Management Tool Survey 2008

The EAMTS 2008 [Ma08], which is the starting point for the evaluation of the open source EA management tool Essential Project, is based on knowledge from the research projects *software cartography* and the successor project *system cartography*. Furthermore, an overview on tools supporting EA management is already provided by the extensive survey of the EAMTS 2005 [se05]. The update of this survey was performed due to increased understanding and importance of EA management as well as the enhanced demands for EA management tool support [Bu08a]. In the course of the EAMTS 2008, the products of nine major players in the market of EA management tools were evaluated: adaptive EAM (adaptive, Inc.), planningIT (alfabet AG), ADOit (BOC GmbH), EA/Studio (Embarcadero), ARIS IT Architect (IDS Scheer AG), MEGA ModelingSuite 2007 (MEGA International SA), Metastorm ProVision (Metastorm), System Architect (Telelogic AB), and Troux (Troux Technologies, Inc.).

The tool survey has been designed in general to support companies that are engaged in EA management endeavors and in particular for companies, which plan to introduce an EA management tool. For this purpose the EAMTS 2008 provides in its introductory chapters and especially in Chapter 3 a profound introduction to the topic of EA management by referring to the previous research of the Enterprise Architecture Pattern Catalog [Bu08b]. Furthermore, the topics of gathering information and integrating EA management with related management areas are delineated. A detailed description of the layers and cross-functions of an EA is given as well as a depiction of possible measures and metrics. Hence, the tool survey can be regarded as a guideline to various

parts of an EA management [Ma08]. The EAMTS 2008 evaluates the investigated EA management tools with respect to best practice scenarios, which were compiled with the aid of and based on the practical experience of the sponsors and partners of this research project. Therefore, the tool survey offers structured decision support for enterprises searching an adequate EA management tool by delineating in detail how a specific tool copes with the different areas and processes related to EA management [Ma08].

The approach of the EAMTS 2008 may be described as a threefold evaluation approach as it relies on two distinct sets of scenarios in combination with an online questionnaire. This questionnaire called list of criteria had to be answered by each of the tool vendors. The first set of scenarios is designed to analyze specific functionality that an EA management tool should provide. It is important to notice that these functionalities are evaluated independently, i.e. without linking them to typical EA management tasks. The second group of scenarios finally investigates the tasks, which were identified as essential parts of many EA management endeavors by the involved sponsors and partners of the research project. In order to ensure the consistency and continuity of the simulation of scenarios, an exemplary data set has been developed. It contains reference data of a fictitious department store, the SoCaStore.

An initial draft of the list of criteria was compiled during the EAMTS 2005. For the EAMTS 2008 the list of criteria was refined, consolidated, and enhanced by the input of the 30 sponsors and industry partner in the course of three extensive workshops, in which also the scenarios were enhanced. The list of criteria constitutes a comprehensive catalog of questions concerning different aspects of EA management tools, such as tool data (e.g. release information, history), tool architecture (e.g. supported platforms, infrastructure requirements), and collaboration support (e.g. locking, collaborative work). The tool vendors answered this list of criteria during an online survey in order to provide additional information to the simulation results.

Besides evaluating the EA management tools against the compiled list of criteria, the two sets of scenarios are simulated. Those scenarios are introduced in detail in a dedicated chapter of the EAMTS 2008. Each of the scenarios is described using the concerns addressed by the scenario and the questions derived from the concerns. Furthermore, the tasks to be accomplished and the deliverables to be created during the simulation of a scenario are depicted in detail. The first scenario for analyzing specific functionality is concerned with *importing, editing and validating model data*. Thereby, the tool's capabilities concerning typical tasks in the importing process are investigated as well as the editing mechanisms for data already contained in the repository. A further scenario determines which types of visualizations (e.g. cluster map, process support map, time interval map, graph layout map, swim lane diagrams) can be created by the tool and to what extent the visualization is generated automatically. The scenario of *interacting with and editing of visualizations of the application landscape* considers for instance the impacts of manual adaptations of visualizations. *Annotating visualizations with certain aspects* tests whether the tool supports the visualization of specific aspects by color-coding or adding traffic lights to symbols in the visualizations. The scenario of *supporting lightweight access* is concerned with evaluating the characteristics of the web access to information and visualizations previously modeled. In the *editing model data using an external editor* scenario the capabilities of the tool for exporting and

subsequently re-importing data as well as the according locking or check-out mechanisms are explored. *Adapting the information model* investigates the flexibility of the information model with respect to adapting, adding, hiding, or deleting attributes, relationships, and classes. The EA management tool's ability in *handling large scale application landscapes* containing up to 10.000 application systems with interconnections is examined. The last scenario of this group is concerned with *supporting multiple users and collaborative work*. Thereby, the type of access control and the according levels of granularity as well as the tool's capabilities for creating notifications and tracking changes in the repository are investigated.

The scenarios for analyzing EA management support depict at first the support for *landscape management*. In this scenario tool's capabilities concerning the modeling of different states of the EA (e.g. current, planned, target) and of landscape variants together with the respective visualizations are tested. The *demand management* scenario looks at the possibilities to document demands and link them to the affected elements of the EA such as project proposals and application systems. In the *project portfolio management* scenario the documentation of project proposals and their links to application systems, business processes, and organizational units are explored. The *synchronization management* scenario addresses issues of (re)scheduling projects with respect to their interdependencies given by the changed objects (e.g. application systems, services). Thus, conflicting projects, which change the same application system at the same time, are detected and furthermore the impacts of a delayed project can be investigated. The scenario of *strategies and goals management* is concerned with the tool's capabilities of aligning projects and demands and therefore also the affected elements of the EA to the defined strategies and goals of an enterprise. In the course of the *business object management* scenario the abilities to model business objects and their flow between application systems are simulated. The scenario of *SOA transformation* explores capabilities to support the enterprise in transforming its architecture into a Service Oriented Architecture (SOA). Thereby, certain characteristics of application systems and services are tested in detail. The *IT architecture management* scenario deals with the introduction and implementation of blueprints and architectural solutions, which are standardizing the architectures of application systems. The last scenario of this group called *infrastructure management* is concerned with issues of the IT infrastructure of the company. Thus, the infrastructure elements (e.g. databases, middleware systems) and their relationships to application systems are determined.

The data needed for simulating the scenarios depicted above should be modeled according to a *meta model*, which in this case is called *information model*. Many of the EA frameworks, which have been proposed in literature make suggestions for organizing an EA. However, no commonly accepted standard information model has yet evolved. Some researchers even doubt the existence of a "one-size-fits-it-all" model [Bu07, KW07]. For this reason, the information model for the SoCaStore was developed for the EAMTS 2008 (see Figure 2). This information model integrates the layers and cross functions as delineated in the previous Section 2.1. Object-oriented modeling is used to provide a common understanding of the information model. Besides the class diagram showing the included EA concepts together with their relationships, a detailed definition of each single class is provided in order to avoid misconceptions.

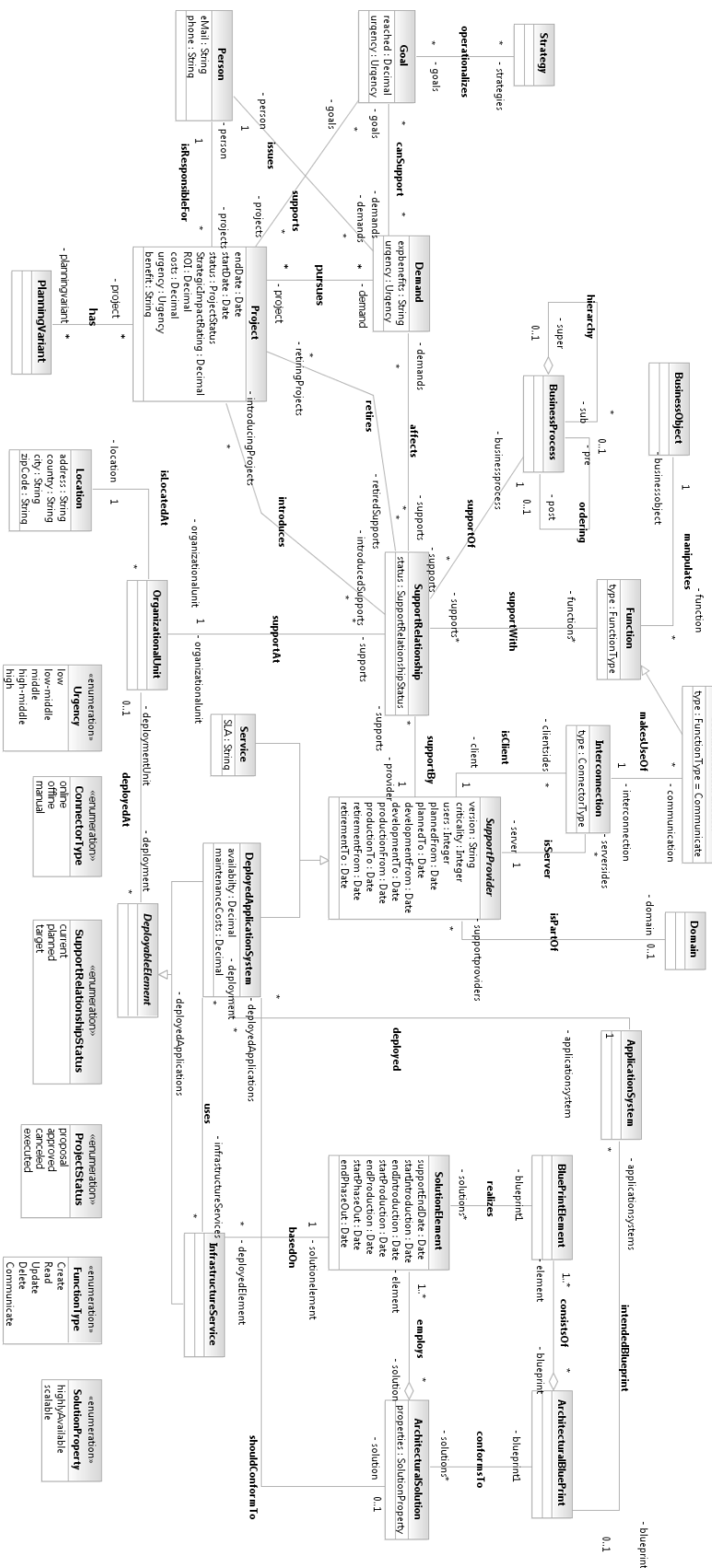


Figure 2: SoCaStore information model. Source: [Ma08]

In the course of evaluating the nine EA management tools, several different approaches of the tools to EA management were detected and incorporated in the executive summaries in Chapter 2 of the EAMTS 2008. The *meta-model driven approach* is pursued by the majority of the evaluated EA management tools. It is characterized by a strong focus on the information model, which can in most cases be described as elaborate and flexible in order to be adaptable to the user's specific requirements. The *process-driven approach* focuses on supporting every typical activity in EA management. EA management tools following a *methodology-driven approach*, such as the ARIS Platform of IDS Scheer AG, adopt industry standard methodologies like the balanced scorecard. The *integration approach* may be characterized by a tool architecture that has been designed to support the aggregation and integration of EA information drawn from various sources.

The evaluation results of the EAMTS 2008 are not compiled into a simple ranking of the investigated tools. Instead, a detailed description of the scenarios is provided for each EA management tool. The results obtained from simulating the various scenarios are also captured in two kiviat diagrams. The first kiviat diagram shows how specific functionalities are covered by the respective tool, while the second diagram reveals the abilities of the tool concerning the support of the EA management tasks investigated in the course of the second set of scenarios. In addition to the two kiviat diagrams for each EA management tool, a concise summary of the findings is given in the EAMTS 2008. This comprehensive approach to presenting the evaluation results enables the readers on the one hand to quickly get an overview of the capabilities of a tool and on the other hand to match their specific requirements with the evaluated EA management tools.

2.3 Open source

Over the past years several successful open source software products have been and still are being developed, distributed, and supported on a voluntary basis by and for users of the software. According to the Open Source Initiative (OSI) it all started with the contribution of Eric Raymond's paper "The Cathedral and the Bazaar" [Ra99], in which he analyzed and described the folk practices in the hacker community. It is said that this paper contributed to the release of the source code of Netscape's popular web browser as free software. The notion of open source was finally coined in February 1998 during a strategy session in Palo Alto[2]. The conference participants felt that the term *free software*, which has so far been used for this kind of software, should be substituted by the newly created label of *open source*. This expression is less ambiguous and therefore companies are more comfortable with it. Subsequently, the OSI was founded and shortly afterwards the initial definition of open source was proposed. According to the definition open source doesn't just mean access to the source code, but that the distribution terms of open-source software must adhere to certain criteria[3]:

[2] http://www.opensource.org/history, accessed 15.08.2010
[3] http://www.opensource.org/docs/osd, accessed 15.08.2010

- *Free redistribution* means that the license does not require royalties or fees for software being a component of an aggregate software distribution.
- *Source code:* The second criterion indicates that the program must include source code or there is a means of obtaining the source code for a reasonable reproduction cost, such as downloading via the internet without charge.
- *Derived works:* The license must allow for modifications and derived works, which may be distributed under the same terms as the license of the original software.
- *Integrity of the author's source code* must be ensured by the license. Therefore, the license must explicitly permit distribution of software built from modified source code. The license may require derived works to carry a different name or version number than the original software.
- *Discrimination against persons and groups:* The license must not discriminate any person or groups of persons.
- *Discrimination against fields of endeavor:* The license may not restrict the software from being used in a specific field of endeavor, i.e. the usage in businesses may not be prohibited.
- *Distribution of License:* The seventh criterion indicates that the rights attached to the software apply to all to whom the program is redistributed.
- *License must not be specific to a product:* The rights attached to the program must not depend on the program's being part of a particular software distribution. In case a part of the program is used or distributed within the terms of the program's license, all parties to whom the program is redistributed should have the same rights as those granted with the original software distribution.
- *License must not restrict other software:* The license must not place restrictions on other software that is distributed along with the licensed software.

The last criterion, which has been added in 2004 declares that the *license must be technology-neutral*. As can be seen from the open source definition, licenses have to comply with the above mentioned criteria and therefore have to go through an approval process. A number of OSI approved licenses is listed on the OSI website[4], amongst which the GNU General Public License can be found, which is one of the best known open source licenses.

Open source software is usually created within an open source project that has been initiated by a group or an individual intending to develop a software product to meet their own needs [KH06]. The open source products are developed by so called open source development communities, which consist of people who contribute by writing code using a few commonly shared coding languages like C++ and Java. Those contributors may be separated into two groups. Firstly, a group called investors, which

[4] http://www.opensource.org/licenses/index.html, accessed 15.08.2010

is made up of programmers who expect to receive rewards from contributing to the open source software. The second group of donators does not expect any reward [FJ02]. When a large number of contributors is developing software, project governance is challenged with forking as there are self-interested contributors who develop their own versions of the software [KH06]. According to [KM01] there is a governance structure in open source software projects, which organizes the work and prevents the software from forking into many version of the original code base. As only light coordination activities performed by a central project team are necessary, no coordination devices beyond mailing lists and versioning software are needed [KH06]. Since the coordination activities are performed by a central team, it influences the product design [KM01]. [AB02] describe two slightly different processes how a contribution can be made. Usually an idea for a change is implemented, tested, and submitted as a patch by a contributor. In the following the implementation is evaluated through testing, reviewing, and discussing until it is rejected or approved by a moderator. If the contribution is made by a trusted developer with write access to the repository, the contributor conceives a change and implements and tests it before he applies his work directly on the repository. The final evaluation may lead to the coordinator reverting the change.

During the last few years a few open source EA management tools have been proposed and receive increasing interest as they feature benefits such as meta-model transparency, openness, no license costs, ability to start small, and the recognition from a growing community[5]. Despite, it must be mentioned that the open source EA management tools currently offer by far less functionality than proprietary tools. Furthermore, there is usually no support offered in case of problems with the open source EA management tool.

In the course of the thesis three open source EA management tools are examined. Iteraplan in its open source version called community edition has been released by the German software and consulting company iterate GmbH in early 2008. This EA management tool has been released under the AGPLv3 license and differs from the community edition in several aspects, such as LDAP connection and Oracle or MySQL support to allow for multi use mode. The second well-known open source EA management Tool is called Essential Project and has been released by the UK-based consulting company Enterprise Architecture Solutions Ltd in March 2009. It is based on the open source ontology editor Protégé[6] and is available under the GNU General Public License. The Essential Project is evaluated in the course of this thesis and is described in detail in Section 3.1. The Tricia/SyCaTool has been contributed by the sebis chair of the Technische Universität München and consists of two tools that have been joined to form an EA management tool. Tricia is an open source Enterprise 2.0 platform, which offers a variety of functionality. In the context of EA management its HybridWiki

[5] http://www.forrester.com/rb/Research/open_source_solutions_for_ea_tool_needs/q/id/55019/t/2, accessed 23.08.2010

[6] http://protege.stanford.edu/

component constitutes the basis for developing an information model and capturing EA information. The SyCaTool is used for creating visualizations of the application landscape. The Tricia/SyCaTool is delineated and evaluated in short according to the scenarios of the EAMTS 2008 in Section 4.2. A comparison of the capabilities of the three open source EA management tools according to the scenarios of the EAMTS 2008 is provided in Section 4.3 of the thesis.

3 Evaluating Essential Project

This chapter of the thesis is concerned with evaluating the open source EA management tool *Essential Project*. It commences with a detailed description of the development, the components, and the tool architecture of *Essential Project*. Section 3.2 describes the adaptations that were necessary to adjust the list of criteria from the EAMTS 2008 to be suitable for open source EA management tools. The last section of this chapter describes the evaluation of *Essential Project* according to the approach taken in the EAMTS 2008.

3.1 The Essential Project

The *Essential Project* was developed by Enterprise Architecture Solutions (EAS) Ltd, a UK-based specialist consultancy in EA. EAS was engaged in a number of projects with companies that wished to apply EA management practices, but were unable to cost justify the investment in a commercial EA management tool[7]. Therefore, EAS selected Protégé, an open source knowledge repository developed by Stanford University, as a starting point for their own EA management tool. Furthermore, EAS developed the *Essential Meta-Model*, which provides enough flexibility to allow a mapping to established frameworks, methodologies, and tools. The Protégé knowledge repository together with the *Essential Meta-Model* and a number of custom extensions makes up the *Essential Architecture Manager*, which constitutes a simple and intuitive means to capture and report on an organization's architecture. EAS applied the *Essential Meta-Model* and the *Essential Architecture Manager* in a number of real-world situations and thus conceived the potential value that these tools could have to the broader community. It was expected that if these tools were available as open source, others would be motivated to develop and share extensions, which would enhance the tool's capabilities and also provide a means for sharing experiences and knowledge among EA practitioners. As a result the *Essential Project* was launched as an open source EA management tool in March 2009.

Although the *Essential Project* is a rather young open source tool, it already has a community of over 670 (June 2010) registered users. Five of those users have already contributed software back to the community and about the same number of users supplies intellectual content to further develop the *Essential Meta-Model*. Thereby, the decision making is governed by a core team that decides which new ideas are introduced to the product. In this context the *Essential Community Process* (ECP) is concerned with capturing requirements, ideas, and proposed solutions as well as the testing of those solutions and the approval for release. The contributed enhancements, extension packs, and new features are released frequently through the community/share area on the website.

[7] http://www.enterprise-architecture.org/about/background, accessed 24.10.2010

The *Essential Meta-Model* is an ontology for the domain of EA that has been developed based on experiences gained from applying EA practices with a variety of EA frameworks and tools. The intention is to provide a comprehensive and extensible set of EA concepts and relationships that can be mapped to the concepts and activities of the already existing EA frameworks. The *Essential Meta-Model* as shown in Figure 3, is partitioned into two segments, the *Core Meta-Model* and the *Support Meta-Model*. The *Core Meta-Model* contains layers, which represent the areas of the EA to be understood. In addition, levels of abstraction, from which this understanding is to be viewed, are defined[8]. The *business layer* contains knowledge that is related to the objectives, capabilities, people, and processes of an enterprise. The functional behavior provided by technology systems while supporting business processes is captured in the *application layer*. The *information layer* comprises knowledge about structured and unstructured information, which supports business processes. Furthermore, this information is managed by application systems and is transmitted or stored using technology. Information about software and hardware technology that is used to implement application systems and to transmit or store information, is kept in the *technology layer* of the *Essential Core Meta-Model.* The *conceptual abstraction view* comprises the capabilities and concepts, which represent the fundamental elements needed to fulfill the objectives of an enterprise. The approaches taken to realize the capabilities and concepts defined in the conceptual view are summarized in the *logical abstraction view*. The *physical abstraction view* contains the implementation of the approaches described in the logical view. The *Essential Support Meta-Model* comprises a number of EA concepts and their relationships, which enable management and governance processes using the knowledge captured in the *Core Meta-Model*. *Strategy management* is concerned with managing a company's future state and the road map for achieving this. The *change management* supports project portfolio management and program management by capturing the dependencies that impact business processes, people and IT during change. The management of the dependencies between business processes, people and IT while supporting the on-going operation of an enterprise is covered by *service delivery*. The parts of the *Support Meta-Model* shown in grey, such as *security*, are planned for development.

[8] http://www.enterprise-architecture.org/component/content/article/11-project-components/35-essential-meta-model, accessed 25.10.2010

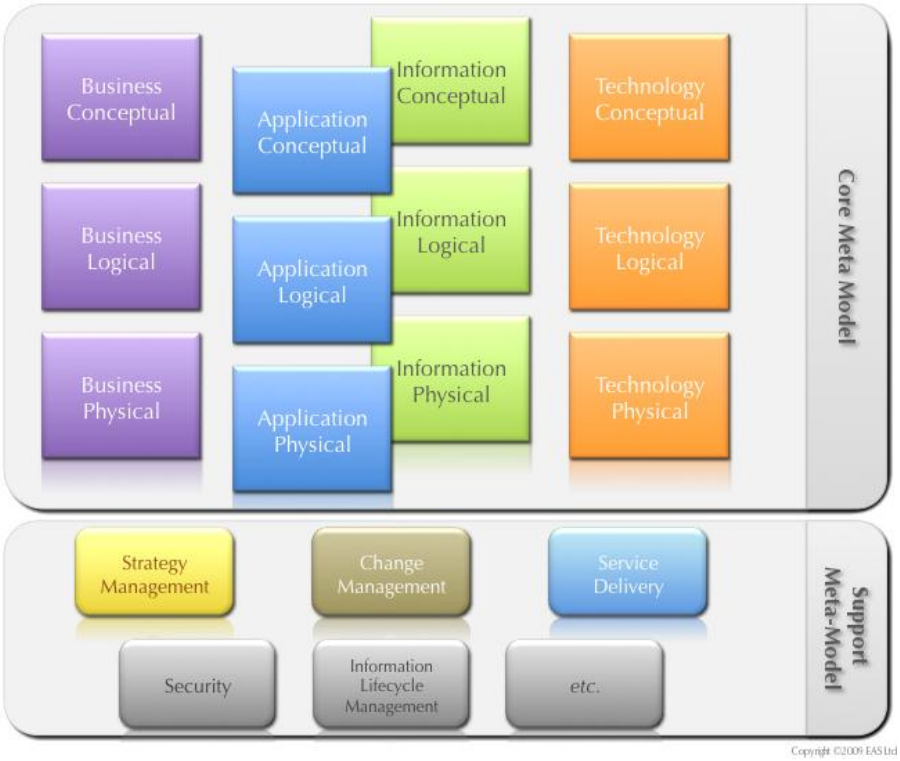


Figure 3: Essential Meta-Model. Source: http://www.enterprise-architecture.org/component/content/article/11-project-components/35-essential-meta-model, accessed 25.10.2010

The *Essential Project* is the collective name for a set of open source tools, which support EA management. Basically, these tools can be separated into two groups. The first group supports the users in modeling an enterprise and the second group provides functionalities for creating views, reports, and performing analyses. This grouping results in the design of the Essential toolset with the two main components that strictly separate capturing information from analyzing it: the *Essential Modeller* and the *Essential Viewer* (see Figure 4). The *Essential Modeller* provides support for capturing and maintaining the EA model. Following the open source philosophy, the *Essential Modeller* is based on the open source ontology editor and the knowledge repository Protégé developed by the Stanford University. Protégé is used to define an ontology, which represents the *Essential Meta-Model*. Furthermore, a number of custom-built Protégé extensions were created so that the *Essential Modeller* constitutes a simple and form-based means of capturing and managing an EA model. Protégé may be supported by a relational database such as MySQL, however this is optional, but recommended when operating the *Essential Modeller* in multi-user mode. The *Essential Viewer* is a java-based web application that can be run on any standard Java server platform (e.g. Apache Tomcat). The information captured with aid of the *Essential Modeller* is published to the *Essential Viewer* by using the custom-built "Essential Architecture Reporting" tab in Protégé. A number of standard reports are predefined and shipped with the *Essential Project*. Nevertheless, the *Essential Viewer* is explicitly designed to enable and encourage enterprises to define and publish their own custom views and reports in order to meet their enterprise-specific needs. Thus, the *Essential Viewer*

constitutes a flexible and extensible means of generating views and reports of the model, which is captured by using the *Essential Modeller*.

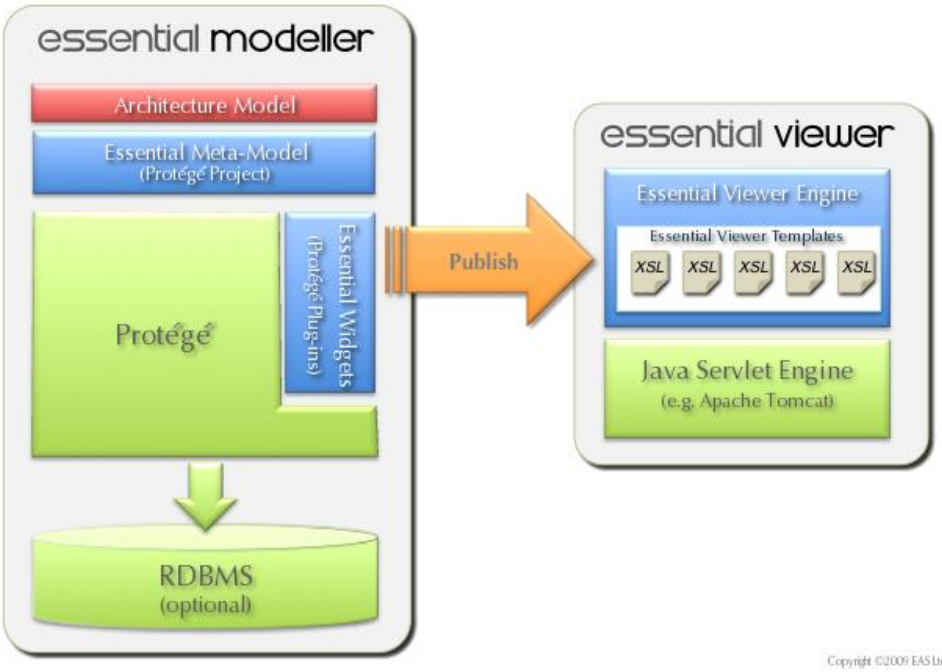


Figure 4: Essential Project software architecture. Source: http://www.enterprise-architecture.org/documentation/doc-administration/36-essential-architecture-manager

3.2 Adapting the list of criteria

The list of criteria defined in the EAMTS 2008 served as a starting point for the adaption of the list of criteria. In several steps this comprehensive catalog of questions was adjusted to the characteristics of open source tools. The final version of the adapted list of criteria can be found in the appendix C. The notion of "tool vendor" was replaced by the term "initiator" as an open source tool is typically not sold by a vendor, but rather developed by community after an initiator has made an initial contribution. Furthermore, a complete section with questions on the community of the open source tool was added. Thereby, the goal was to characterize the community with respect to the number of community members, active community members, decision processes, and support of community member among others. In the tool data section an additional question regarding the license under which the tool is available was posed. The other sections of the list of criteria were borrowed from the EAMTS 2008 without further adaptations.

The final version of the list of criteria (see Appendix C) was sent to the *Essential Project Team*, who provided detailed answers to the various questions. The results from the answered catalog of questions were incorporated in the evaluation of the scenarios where appropriate. The characterization of the *Essential Project* community can be found in the previous Section 3.1.

3.3 Evaluating scenarios

3.3.1 Scenarios for analyzing specific functionality

3.3.1.1 Importing, editing, and validating model data

The *Essential Project* provides two extensions to the *Enterprise Architecture Manager* for importing information from external sources, the *Essential Integration Server* and the *Essential Integration Tab* (see Figure 5). Both extensions require the source data to be represented in XML and the transformations are defined by using XSLT. Employing such a script-based approach provides a powerful, flexible, and extensible platform for integration, enabling transformations to be defined and managed between the Essential Meta Model and an external repository. The newer *Essential Integration Tab* is more convenient as, for example, downloading and running the created scripts is done automatically. At the time of the evaluation, there was little documentation available on importing data with either extension. A tutorial on writing integration transforms has been contributed lately, which shows that the fairly young *Essential Project* is constantly evolving. In case assistance for adapting the importing process is required, the Essential Project Team offers professional service for developing custom imports.

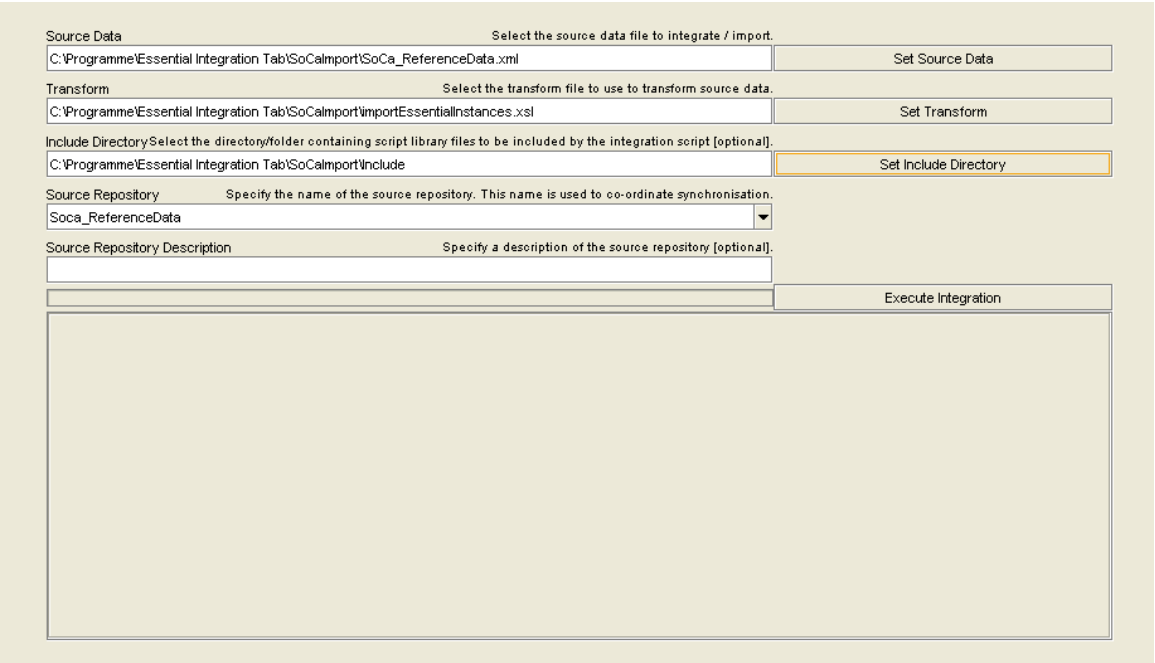


Figure 5: *Essential Project* – Essential Integration Tab

In addition to mere importing the *Essential Integration Server* and *Essential Integration Tab* also provide means to synchronize the information held in the Essential Modeler with external information sources. This functionality may be used for importing regular updates of parts of the architecture, for capturing changes, and for introducing new elements. As several external information sources may be used, the attribute “external repository instance reference” was introduced in order to keep track of the external

repository used. Additionally, each run of the integration timestamps the new and the updated instances in the Essential repository.

As the *Essential Project* pursues a XML/XSLT based approach for importing data from external sources, the splitting of comma separated values in the integration process is possible. However, this must be included in an appropriate XSL file. The imported data is validated during the import process according to aspects like the type or length of an attribute value and in case of a violation an error message is prompted in the console. In case of missing values for mandatory attributes no warnings during the importing process exist. However, when the instance is later viewed in Protégé, a red rectangle appears indicating the missing value for the attribute. If a value is omitted for the mandatory name attribute, the according instance is created without warnings, but it does not appear at all in Protégé.

For editing the information governed in the repository of the *Essential Modeler*, the tool provides input forms, which are automatically adapted to match the information model. Additional input fields or dropdown boxes are added at the time further attributes are introduced. The data in the repository can be accessed using the tree-like navigator of the Protégé class browser (see Figure 6), which provides a comprehensive overview of the information stored in the repository. In *Essential Project* each instance is edited individually and there are no wizards guiding the editing process. Further supportive functions for editing are initializing attributes with default values and pick lists for predefined enumerations and for recording associations. Finally, it is possible to link external documentation, such as Microsoft Word documents, to each instance in the repository by specifying the according paths and file names in the “external reference links” attribute.

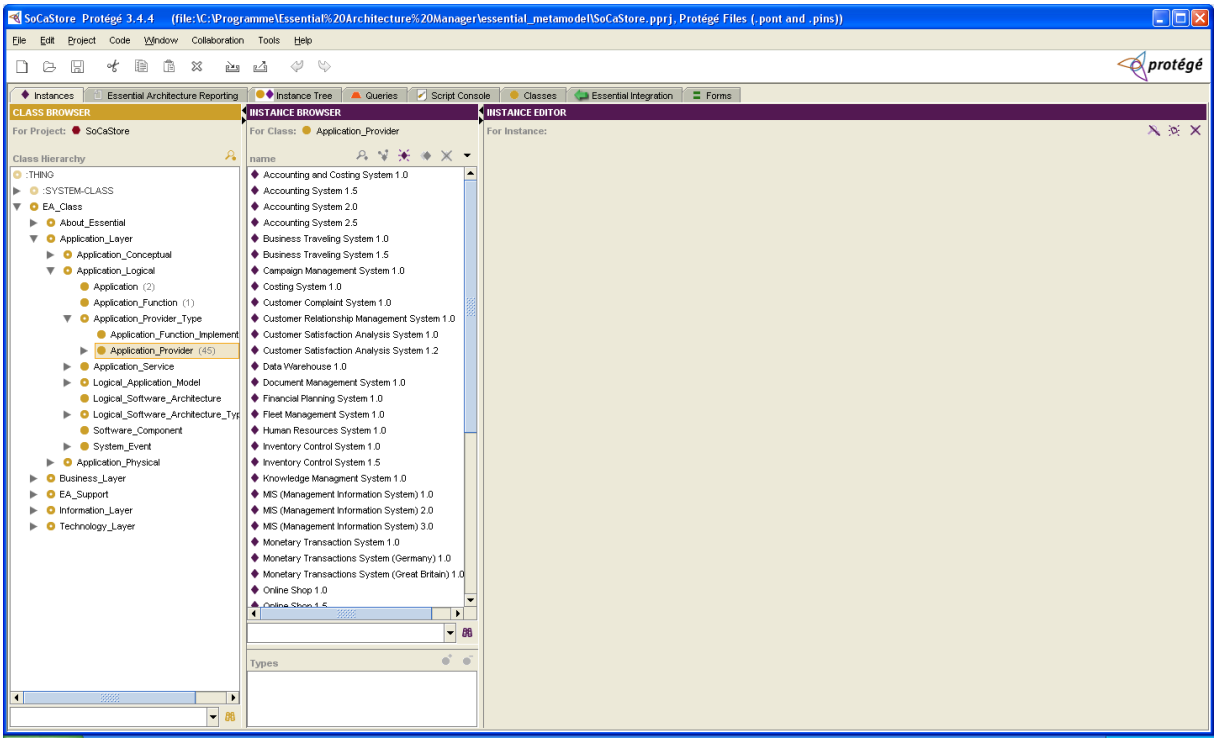


Figure 6: *Essential Project* – class browser navigation

3.3.1.2 Creating visualizations of the application landscape

The *Essential Project* provides a few visualizations, which are *graph layout maps* and *cluster maps* according to the terminology of the EAMTS 2008. Some of these visualizations are created automatically while importing or editing repository data. Figure 7 shows an example of a graph layout map, which has been generated automatically during importing.

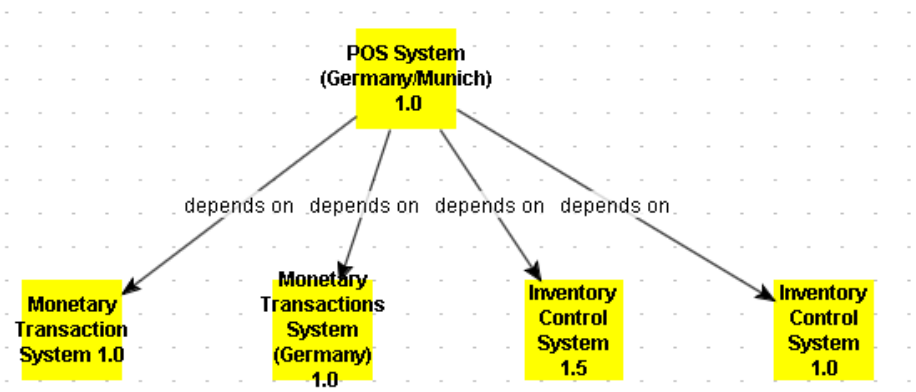


Figure 7: *Essential Project* – automatically generated graph layout map

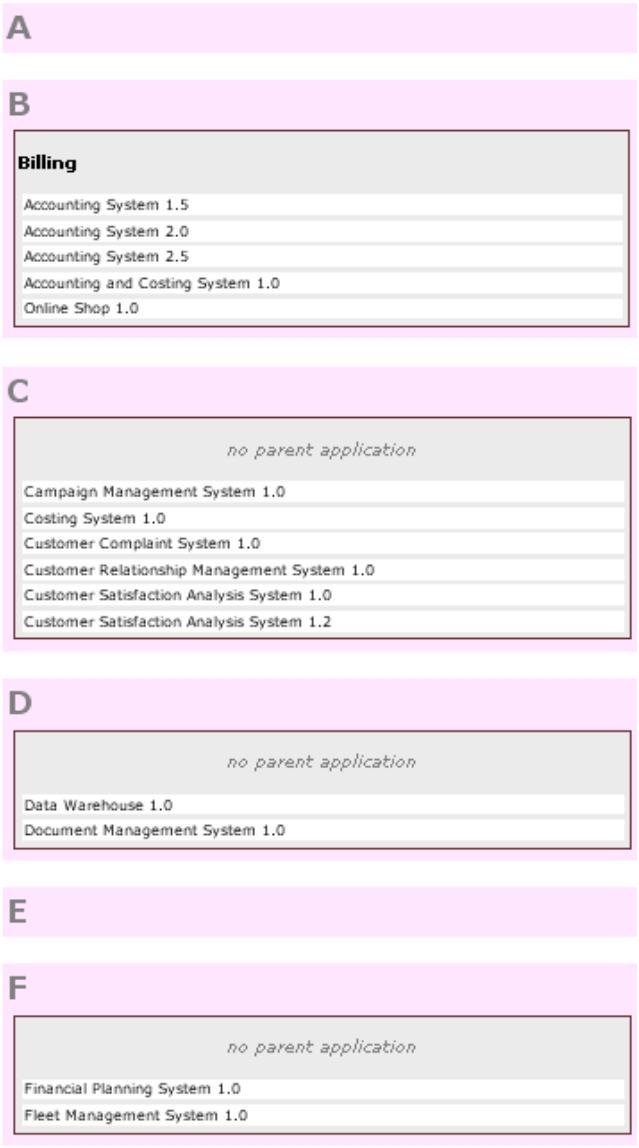


Figure 8: *Essential Project* – excerpt from an automatically generated cluster map

While publishing the Essential Modeler repository, cluster maps are generated and may be accessed via the Essential Viewer. **Fehler! Verweisquelle konnte nicht gefunden werden.** shows a map clustering application systems into domains. Thereby, the elements in the cluster map are arranged alphabetically and those application systems without associated domain are clustered according to the first letter. *Portfolio matrices*, *process support* and *time interval maps* cannot be created by Essential Project as the Essential information model does not provide the required information. Unlike the information model of the EAMTS 2008, the Essential information model does not propose a ternary relationship between application systems, business process and organizational units. This ternary relationship however, is the basis for the correct placement of the application systems on the business process and organizational unit axes. For a *time interval map* the start and end dates of the lifecycle phases are required. This is not covered by the Essential information model and instead it is proposed to assign a single predefined lifecycle phase to an application system, which represents the

current lifecycle phase. Therefore, *time interval maps* cannot be created on the basis of the out-of-the-box information model. However, the Essential information model could be extended with the appropriate attributes. The generation of *portfolio matrices* requires several attributes for projects, which are not part of the Essential information model. *Swim lane diagrams* as demanded in the scenario cannot automatically be created. The exporting functionality provided by *Essential Project* may be used to export appropriate information to Microsoft Excel and thus achieve the creation the different visualizations.

A Visio export tool, which has been contributed by a community member, enables the contents of the Essential repository to be exported to and viewed in Microsoft Visio (see Figure 9). The contribution consist of a python file, which is run with the Protégé script console and creates a VBA script that is then used to generate the visualization in Microsoft Visio. Different contents of the visualization may be chosen by specifying either which layer of the Essential information model or which abstraction view is to be visualized.

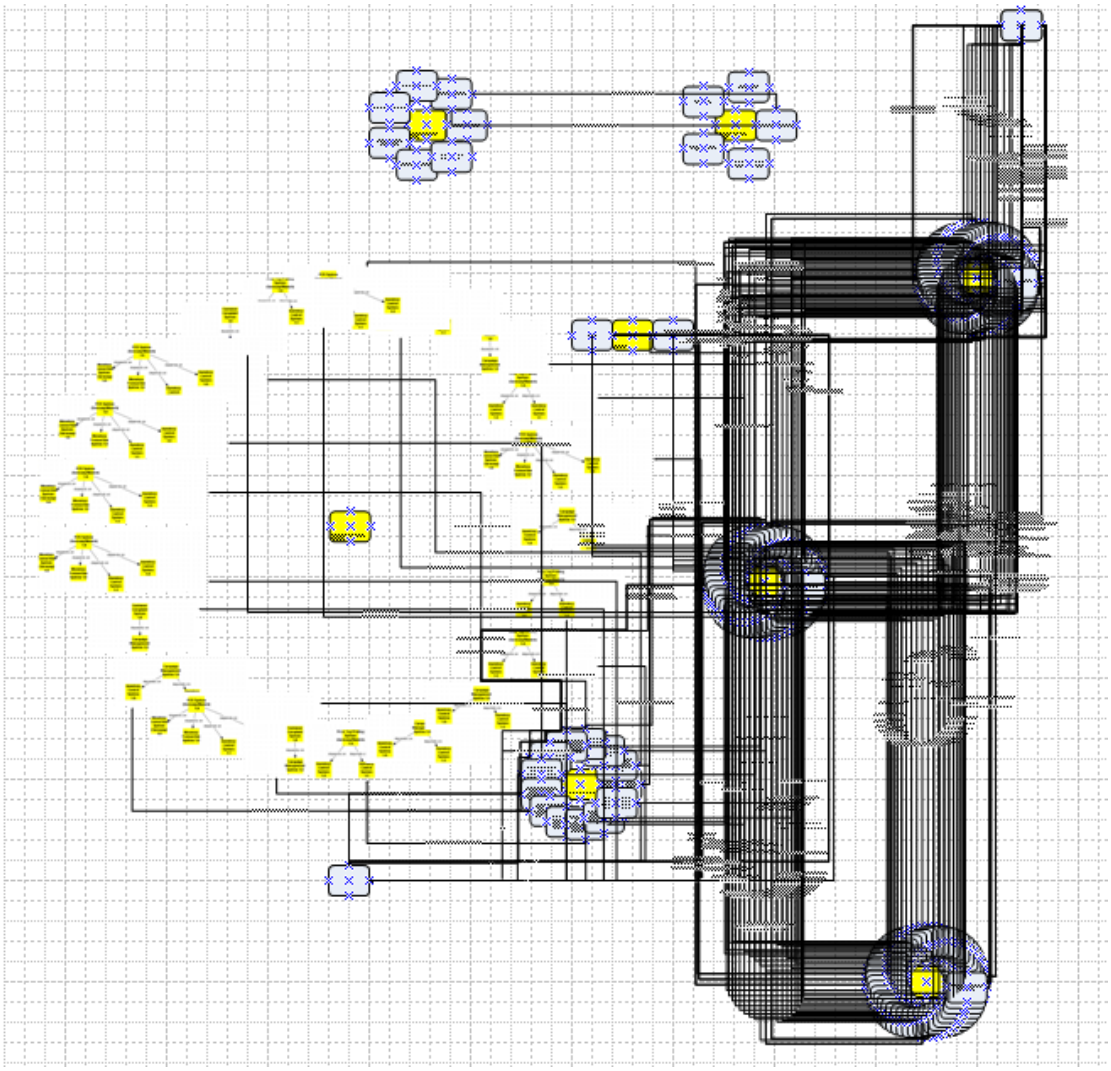


Figure 9: *Essential Project* – graph layout map showing contents of the application layer

3.3.1.3 Interacting with, editing of, and annotating visualizations

Concerning the interaction with visualizations, the three different kinds of obtaining visualizations with *Essential Project* have to be considered separately. At first, we regard the visualizations that are created on importing or editing repository data. Those visualizations are automatically generated graph layout maps representing the current state of the information repository and can only be edited in Protégé. Thereby, the layout of the symbols representing objects from the repository may be changed either manually or by using the automatic layout buttons. Furthermore, symbols and associations between symbols may be added. However, it must be kept in mind that the visualization's semantics are altered concurrently. The visualizations generated on importing or editing do not provide further editing functionalities such as highlighting symbols, filtering or hiding complex inner structures. Furthermore, annotating the visualizations with certain aspects, such as color coding applications with respect to their operating costs or adding traffic light symbols, is not supported.

The visualizations created during publishing the repository content to the Essential Viewer are not intended to be edited as they represent the current state of the information repository. Nevertheless, this kind of visualization provides navigation capabilities, which enable the user to navigate to related objects or to view more details on a certain object.

The third kind of visualization provided by the Microsoft Visio export extension is the most flexible of the three as the full-scale flexibility of Microsoft Visio can be used for changing the visual make up of the visualizations. Thus, the user can freely adapt the visualizations by moving, resizing, adding, deleting, highlighting, changing shape or color of the contained symbols. As the generated Microsoft Visio file is decoupled from the information repository, the adaptations in Microsoft Visio do not exert changes in the underlying repository.

3.3.1.4 Supporting lightweight access

Web access to the information modeled in the *Essential Modeler* is obtained by publishing the repository content to the *Essential Viewer*, which is a java-based web application that runs on any standard Java server platform (e.g. Apache Tomcat). The *Essential Viewer* (see Figure 10) as the name suggests was merely designed for viewing the repository content and therefore no editing capabilities are included.

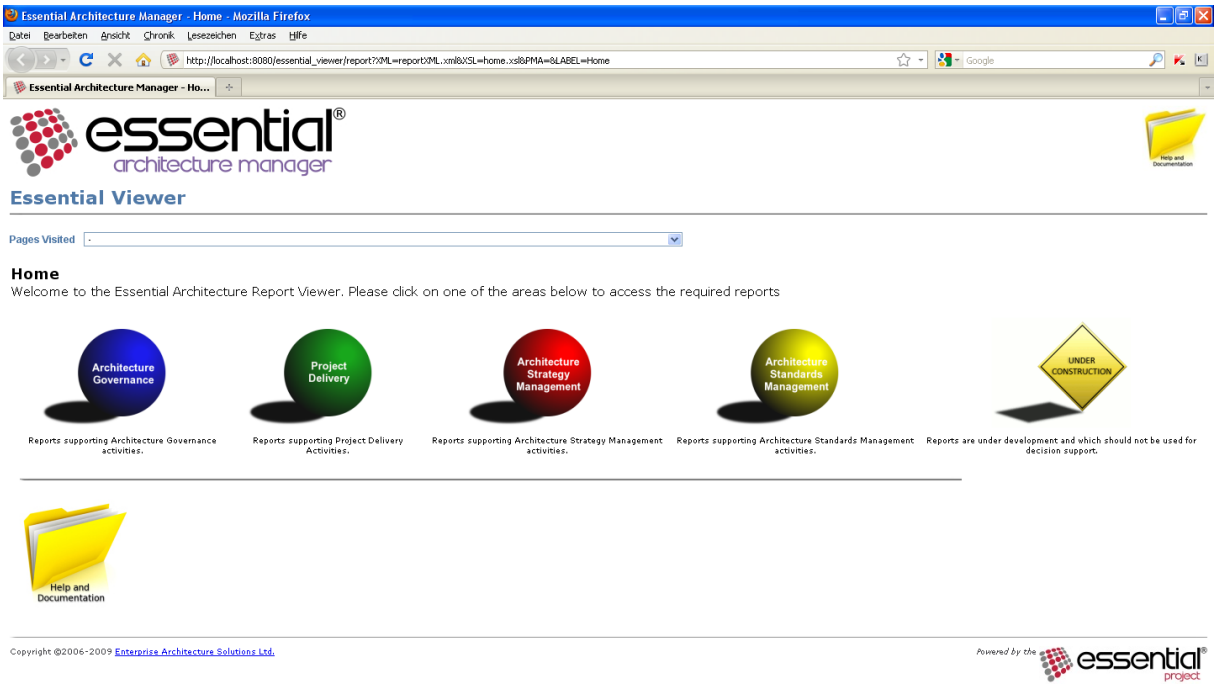


Figure 10: *Essential Project* – Essential Viewer

Access to the *Essential Viewer* can be controlled via the user account, which in turn is controlled by the application server platform. Thus, the overall access to the published content is limited. If only certain visualizations and information may not be viewed by certain users or user groups an additional instance of the *Essential Viewer* can be deployed containing a different set of visualizations, reports, and repository content.

The *Essential Viewer* offers navigation capabilities, which enable the user to get from an overview diagram to the detailed tabular description of a repository object by clicking on the name of the symbol in an overview visualization. The search for objects in visualizations or tabular reports is realized by using the search functionality that comes with the web browser. Advanced capabilities such as showing/hiding layers in visualizations, filtering or hiding certain aspects on the go are not provided by the *Essential Viewer*.

3.3.1.5 Editing model data using an external editor

The *Essential Project* offers a flexible way of exporting repository data for external editing and like the importing process this approach is again based on XSLT. An appropriate xsl file has to be written that produces the required information in the required format, e.g. XML, CSV etc. This xsl file is then specified as the entry point for a report, which displays the desired information in the browser once the *Essential Modeler* repository has been published to the *Essential Viewer*. A contributed extension to the Essential Viewer enables the information to be downloaded by specifying additional URL parameter instead of downloading it from the browser. According to the Essential Project Team it is also planned to produce advanced spreadsheet exports in Excel format.

The Essential Architecture Manager does not provide locking or check-out mechanisms. Thus, data that has been previously exported cannot be prevented from being edited in the information repository. The Essential Architecture Manager, as it is based on the Protégé platform, operates a live repository approach so that each change made to the contents of the repository is committed immediately. This also applies when the data that has been edited externally is re-imported with the Essential Integration Server or the Essential Integration Tab as described in the first scenario. Therefore, it must be checked prior to starting the re-importing process, which of the instances have been altered in the repository after exporting. Otherwise, these changes would be lost as the instances are overridden by the re-imported instances.

3.3.1.6 Adapting the information model

The information model provided by the *Essential Project* is an ontology for the domain of EA and is intended to be a comprehensive and extensible set of concepts and relationships with clear semantics. The *Essential Modeler* is based on the open source ontology editor Protégé, in which the information model is captured. Thus, the information model is very flexible and can be customized to the user's needs. However, it is recommended to take an approach that extends the information model rather than an approach that changes it. Future updates to the information model or to reports and visualizations can be still applied to an extended information model. If the information model is changed substantially future updates can be difficult to apply.

For adapting the information model the classes tab and the form tab of Protégé have to be enabled. The classes already contained in the Essential information model can be deleted, if no more instances for the class to be deleted exist. Furthermore, classes can be hidden, or new classes may be created. Similarly, attributes and relationships may be created, hidden and deleted. Protégé supports the creation of mandatory attributes by checking "required" (see Figure 11) as well as setting default values. If an attribute with already assigned values is deleted or adapted, there is no notification to the user that those values might be lost. The same behavior for deleting and adapting applies to relationships. Default values may also be specified for relationships, however, it is not possible to add attributes to an relationship.

Essential Project supports a number of data types including the basic data types of Boolean, Integer, Float, String. Furthermore, an "Any" data type is supported as well as "Class" and "Instance". From this listing of data types can be seen that the "Date" type, which is essential for EA management is missing.

Changes to the information model are automatically made visible within the graphical user interface of the *Essential Modeler*. Furthermore, every user interface form of a class can be customized in terms of the layout and the labeling of the fields by making the according alterations on the form tab (see Figure 12).

The Essential Project does not provide functionality to visualize the current information in a class diagram. Nevertheless, Protégé provides several ways for exporting the current information model either on its own or together with the contents. In the course

of the evaluation the RDF Schema export functionality was tested, which provides the user with two files. A RDF Schema file containing the information model and a RDF file representing the repository content. With the aid of these two files the Protégé project can be recreated by re-importing them.

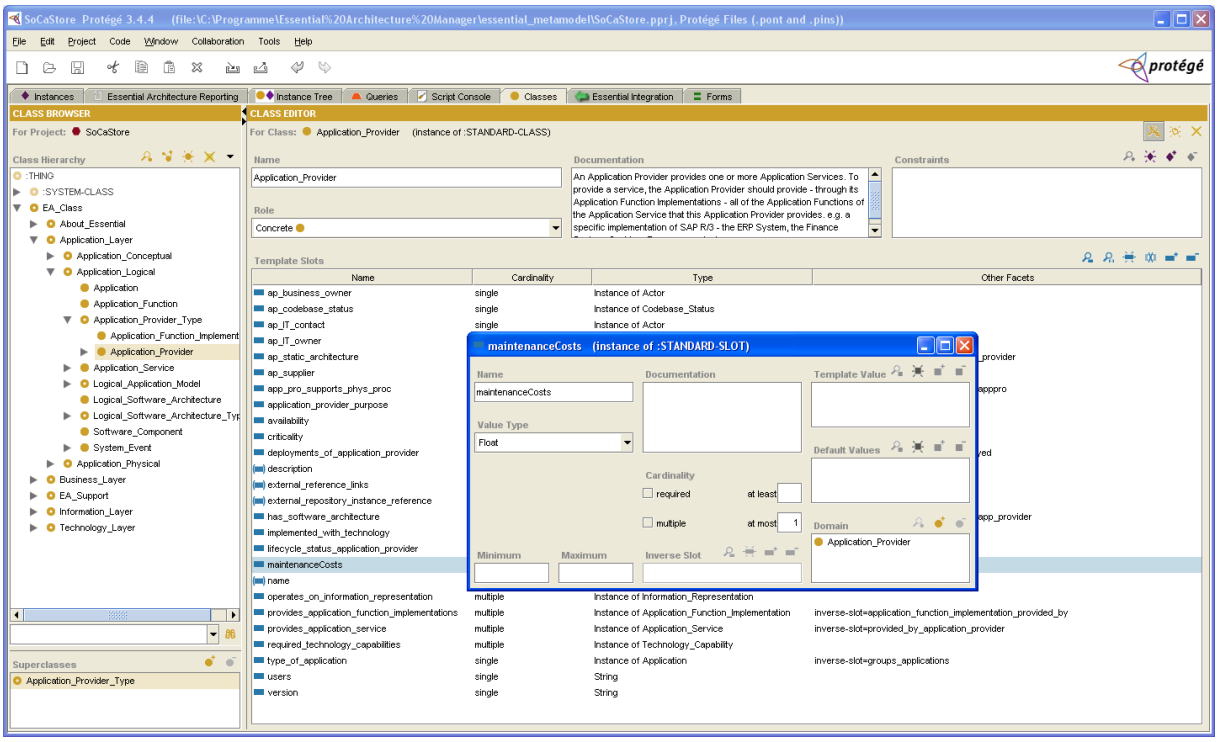


Figure 11: *Essential Project* – adding an attribute

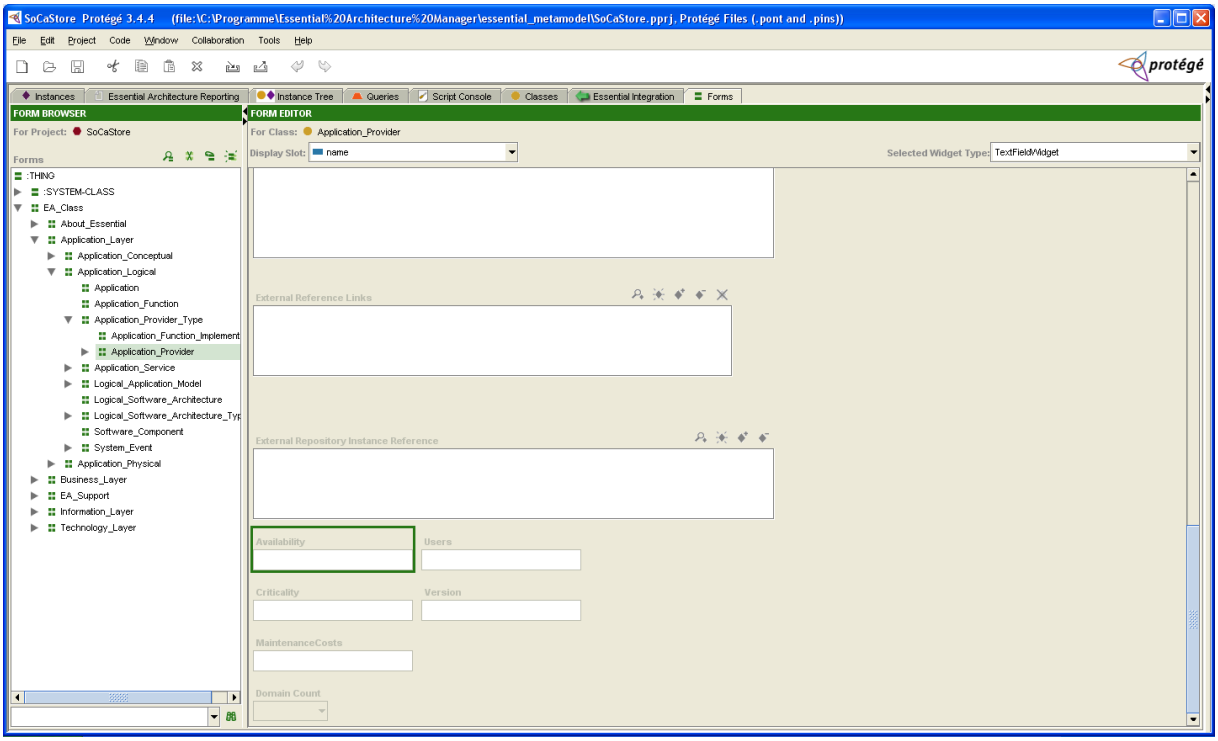


Figure 12: *Essential Project* – customizing the user interface via the form tab

3.3.1.7 Handling large scale application landscapes

Importing large scale data into the Essential Modeler via the Essential Integration Tab does not cause any problems. Nevertheless, the importing of the 10000 application systems took longer than a *coffee break* duration when speaking in the terminology of the EAMTS 2008 for perceived duration times, but not *overnight*.

The *Essential Project* provides in fact some supporting functionalities to deal with large scale data, such as ordering the instances alphabetically by their name in order to ease the navigation within the instance browser. Another example for a supportive functionality of the *Essential Project* in the context of large scale data is the search functionality implemented by the queries tab. This enables the user to search for instances that fulfill the criteria specified for their attributes. Furthermore, the interaction with the *Essential Modeler* seems not to be affected by the large number of instances in the repository.

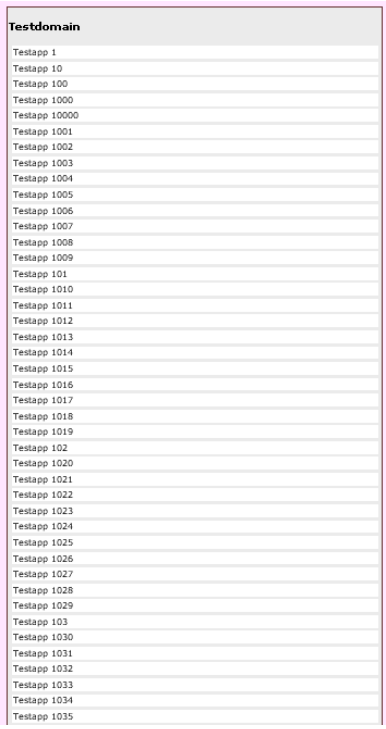


Figure 13: *Essential Project* – excerpt from a cluster map containing 10000 application systems

Publishing the repository containing large scale data to the *Essential Viewer* does not take considerably longer than with the previous amount of data although all the visualizations and reports include significantly more elements. Working with the *Essential Viewer* containing large scale data reveals no substantial impact of the large amounts of data on the performance of this web-based application. The *Essential Viewer* is capable of producing the cluster map for 10000 application systems as shown in Figure 13. The Microsoft Visio export extension on the other hand cannot cope with large scale data. Even with only 1000 application systems it is not possible to obtain the VBA script for creating the according Microsoft Visio visualization as a Java *OutOfMemoryError* exception is thrown.

3.3.1.8 Supporting multiple users and collaborative work

The *Essential Project* supports multiple users as well as collaborative work. A relational database is recommended when operating the Essential Project in multi-user mode, although this is optional. An installation of the *Essential Modeler* is deployed on a server and on the workstation of each user who is updating the contents in the repository. The Protégé platform that underpins the *Essential Modeler* operates in an online mode so that each change, which is made to the contents of the repository, is committed immediately. For this reason, regular repository backups are recommended in order to be able to roll back to the previous backup.

Exploring the provision of access rights is of focal interest when evaluating the capabilities of *Essential Project* for supporting multiple users. Thereby, a distinction must be made between the *Essential Modeler* realized by Protégé and the *Essential*

Viewer. Protégé provides user rights management that applies to the repository as a whole. Thus, a user has either read/write access to the entire repository or is restricted to read-only access. Restricting the access to specific instances or attributes in the repository is currently not supported. Access to the *Essential Viewer* is controlled by user account by the application server platform Tomcat. For preventing certain users from viewing certain types of reports or visualizations it is recommended to deploy additional instances of the *Essential Viewer* containing different sets of views and reports.

The *Essential Modeler* which is based on Protégé provides powerful collaboration capabilities, which allow multiple users to cooperate and agree upon the knowledge held in the repository. With this Protégé built-in functionality it is possible to track all the changes made to the repository. In addition, annotations of various types, such as comment, advice or example, can be added to classes and instances in the repository (see Figure 14). A further feature provided by the collaboration capability is the live chat with other users that are logged on the same Protégé server.

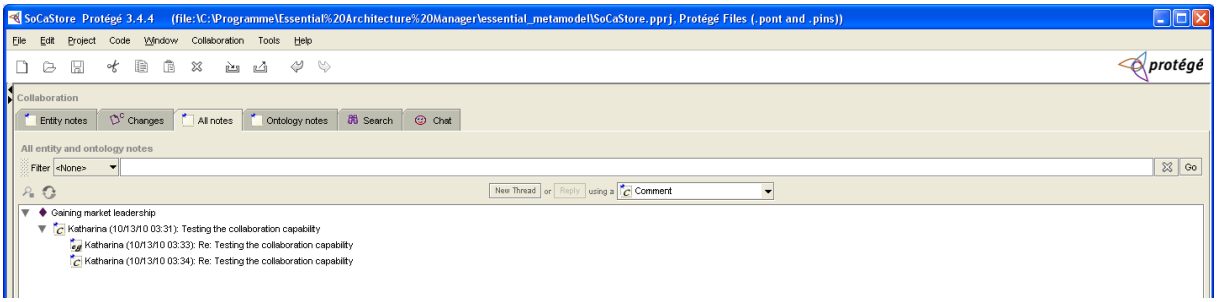


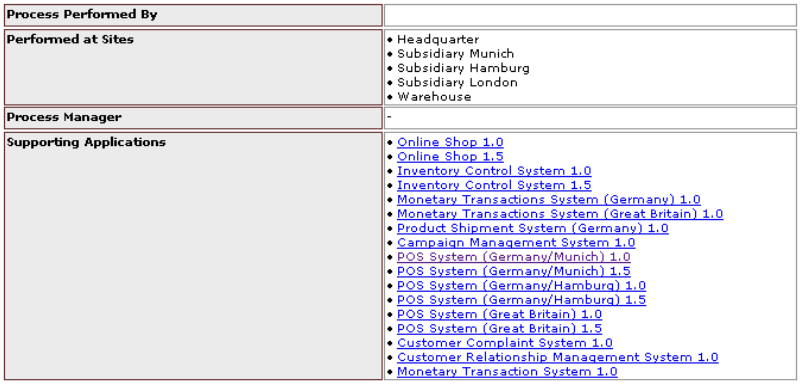
Figure 14: *Essential Project* – annotating an instance using the collaboration capability of Protégé

3.3.2 Scenarios for analyzing EA management support

3.3.2.1 Landscape management

The Essential information model contains a number of classes to represent the necessary objects, which make up an application landscape. For instance, the information model comprises classes to model application systems, business processes, and organizational units. However, the relationships are defined in a different way than in the *SoCaStore* information model, which proposes a ternary relationship to express that an application system supports a business process at an organizational unit. The Essential information model instead proposes that an application system supports a business process, which is located at a certain site or organizational unit. Therefore the information model does not provide for the creation of process support maps, which are the central type of visualization for the scenario of landscape management. As an alternative, either the application module specification or the business process specification (see Figure 15) published in the *Essential Viewer* may be used. A major drawback however is that this has to be viewed individually for each instance of business process or application

system as no visualization or report is provided that contains the entirety of business processes and application systems. Nevertheless, a custom report may be defined for this purpose.



Process Performed By	
Performed at Sites	• Headquarter • Subsidiary Munich • Subsidiary Hamburg • Subsidiary London • Warehouse
Process Manager	-
Supporting Applications	• Online Shop 1.0 • Online Shop 1.5 • Inventory Control System 1.0 • Inventory Control System 1.5 • Monetary Transactions System (Germany) 1.0 • Monetary Transactions System (Great Britain) 1.0 • Product Shipment System (Germany) 1.0 • Campaign Management System 1.0 • POS System (Germany/Munich) 1.0 • POS System (Germany/Munich) 1.5 • POS System (Germany/Hamburg) 1.0 • POS System (Germany/Hamburg) 1.5 • POS System (Great Britain) 1.0 • POS System (Great Britain) 1.5 • Customer Complaint System 1.0 • Customer Relationship Management System 1.0 • Monetary Transaction System 1.0

Figure 15: *Essential Project* – excerpt from the business process specification of distribution

For planning applications landscapes it is further important to be able to collect data regarding the evolution of the landscape. It should be possible to gather information for the current, planned, and target states of the application landscapes in order to generate appropriate visualizations of those states. However, this time dimension is not part of the Essential information model, which merely considers the current state of the application landscape. Instead of recording the time intervals for the different lifecycle phases of an application, *Essential Modeler* only allows to assign a single lifecycle phase, which represents the current phase, to an application system. As only the modeling of the current state of the application landscape is supported, is it further not possible to model different versions of the application landscape and thus different planning scenarios cannot be envisioned.

A further drawback for planning application landscapes is that the Essential information model provides no direct relationship of projects to the affected application systems. Projects in Essential do not have an impact on applications, but on the equivalents of architectural solutions and solution elements in terms of the EAMTS 2008 terminology. Thus, the impact of a project on an application system cannot be modeled as suggested by the scenario of landscape management.

3.3.2.2 Demand management

The Essential information model provides no direct support for demand management as the concept of demands is not referred to. Although the Essential information model provides the *project* class for modeling projects, there is no appropriate class related to the *project* class, which could be used for modeling demands. In order to overcome this deficiency the functionalities described in the "Adapting the information model" scenario may be employed to add a *demand* class and the respective relationships to projects and application systems. After adjusting the form for *demand* via the form tab, information for demands may be captured in the *Essential Modeler* as shown in Figure 16.

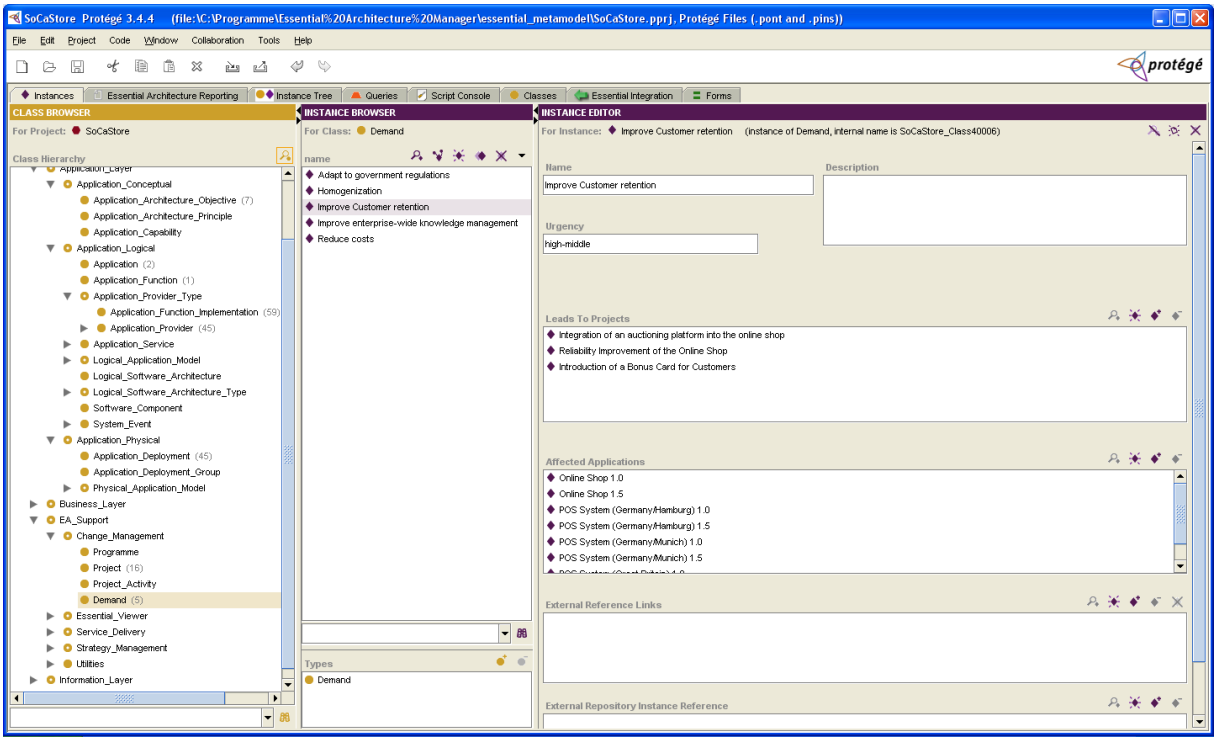


Figure 16: *Essential Project* – example instance of the newly created *demand* class

In order to report on the extended information model, a new report has to be created. This can be achieved by specifying an XSL file producing the desired outputs. Afterwards a new instance in the *report* class of the *Essential Modeler* has to be created and linked to the appropriate report classification and report group. Using this approach the visualization of demands with affected application systems and implied projects that is shown in Figure 17 may be generated.

Demands	Affected Applications	Leads to Projects
Adapt to government regulations	Accounting System 2.0	VAT change in Accounting System
	Accounting System 2.5	VAT change in Costing System
	Accounting and Costing System 1.0	VAT change in Online Shop
	Online Shop 1.0	VAT change in POS System
	Online Shop 1.5	VAT change in Price Tag Printing System
	POS System (Germany/Hamburg) 1.0	
	POS System (Germany/Hamburg) 1.5	
	POS System (Germany/Munich) 1.0	
	POS System (Germany/Munich) 1.5	
	POS System (Great Britain) 1.0	
	POS System (Great Britain) 1.5	
	Price Tag Printing System (Germany/Hamburg) 1.0	
	Price Tag Printing System (Germany/Hamburg) 1.5	
	Price Tag Printing System (Germany/Munich) 1.0	
	Price Tag Printing System (Germany/Munich) 1.5	
	Price Tag Printing System (Great Britain) 1.0	
	Price Tag Printing System (Great Britain) 1.5	
Homogenization	Accounting System 2.5	Connection of Costing and Accounting System
	Costing System 1.0	Database Consolidation
	Monetary Transaction System 1.0	
	Monetary Transactions System (Germany) 1.0	
	Monetary Transactions System (Great Britain) 1.0	
Improve Customer retention	Online Shop 1.0	Integration of an auctioning platform into the online shop
	Online Shop 1.5	Introduction of a Bonus Card for Customers
	POS System (Germany/Hamburg) 1.0	Reliability Improvement of the Online Shop
	POS System (Germany/Hamburg) 1.5	
	POS System (Germany/Munich) 1.0	
	POS System (Germany/Munich) 1.5	
	POS System (Great Britain) 1.0	
	POS System (Great Britain) 1.5	
Improve enterprise-wide knowledge management	Customer Satisfaction Analysis System 1.0	Introduction of Knowledge Management
	Customer Satisfaction Analysis System 1.2	Introduction of a new Management Information System
	Knowledge Managment System 1.0	Replacement of the Customer Satisfaction Analysis System by the Data Warehouse
	MIS (Management Information System) 2.0	
	MIS (Management Information System) 3.0	
Reduce costs	Inventory Control System 1.0	Consolidation of Monetary Transaction Systems
	Inventory Control System 1.5	Introduction of RFID in the Warehouse
	Monetary Transaction System 1.0	Optimization of Monetary Transaction System
	Monetary Transactions System (Germany) 1.0	
	Monetary Transactions System (Great Britain) 1.0	

Figure 17: *Essential Project* – custom report for showing collected demands and their relationships to projects and application systems

3.3.2.3 Project portfolio management

In the context of this scenario, similar considerations as in the previous scenario have to be taken, as the *Essential Project* actually supports projects, but they are not directly linked directly application systems in the predefined information model. Instead a project is modeled to change architectural solutions (*Essential:* technology product build) and solution elements (*Essential:* technology product). Therefore, two additional relationships to application system, indicating the introduction or retirement of an application system by a project, have to be introduced in order to facilitate the evaluation of this scenario.

To get an overview about, which project proposals have been received, the queries tab of Protégé may be used. A query restricting the *status* attribute to the value of proposal can be employed to separate project proposals from projects in other statuses (see Figure 18).

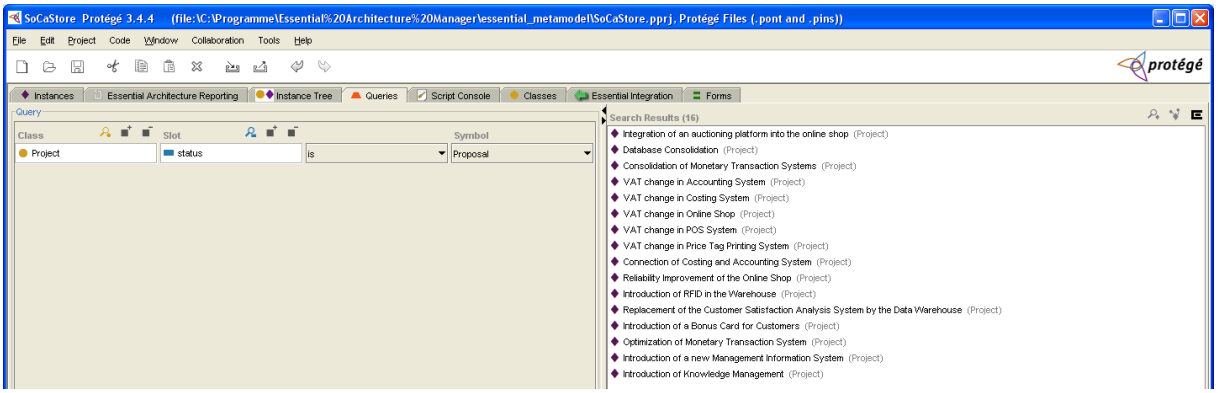


Figure 18: *Essential Project* – query for selecting project proposals

The flexibility of the Essential information model facilitates that the user could annotate, which project proposal have to be accomplished in any case. In order to achieve this, an appropriate Boolean attribute has to be added to the already existing attributes of the *project* class. The Protégé queries tab can subsequently be used to retrieve the set of projects that have to be accomplished in any case. Furthermore, this query can be combined with the one described above, which aims at selecting project proposals.

For analyzing which application systems are affected by a selected project, a manual approach has to be taken as no predefined reports exist for this purpose due to the fact that the relationships between projects and application systems were added. As an alternative to the manual approach, a custom report may be defined in a similar fashion as in the previous scenario.

Deriving potential conflicts in the project portfolio due to the same application system being affected by more than one project at the same time is not achievable as the *start date* and *end date* attributes of project are of type string. A possible solution for this shortcoming is to export the respective data, for instance to Microsoft Excel, and format the *start* and *end date* attributes as date in order to facilitate the detection of conflicting projects.

In order to provide for the creation of project portfolio matrices, several attributes have to be added to the *project* class of the Essential information model. The project portfolio matrix, which is to be evaluated in this scenario, is based on project costs, expected return on investment (ROI), urgency, and strategic impact rating. Hence, the set of attributes of the *project* class was extended by those four attributes. However, the *Essential Project* does not provide functionalities to generate the project portfolio matrix automatically. Again, an export of the required data to Microsoft Excel and the subsequent visualization with the Microsoft Excel capabilities for creating bubble diagrams is a possible way to overcome this limitation.

3.3.2.4 Synchronization management

The Essential Project provides only limited support for the scenario of synchronization management. Project dependencies resulting from affecting the same application system or the same organizational unit at the same time can be derived by performing manual impact analyses. These analyses can be executed by starting with the projects, which should be examined, and traversing along their relationships to the assigned application systems and subsequently via the business processes to the allocated organizational units (*Essential:* Site). More automation concerning this deduction of the dependencies is hard to achieve as the *start* and *end dates* of projects are of type string. Yet, a custom report containing all projects together with their assigned application systems, business processes and organizational units may be defined. Since such a report provides a glimpse on all associations at a time, it is much more convenient than traversing the relationships in the *Essential Modeler*. Other project interdependencies than those derived from projects affecting the same application system or organizational unit at the same time, cannot be modeled according to the Essential information model.

The automatic generation of the different time interval maps as suggested in the scenario of synchronization management is not supported by *Essential Project*. This is in part due to the project *start* and *end dates* being modeled as string attributes. For this reason, the timeframes, in which an organizational unit or application system is affected by a project, cannot be directly derived from the repository data. Nevertheless, a custom tabular report containing the gathered information about project delays may be created as describe in previous scenarios.

3.3.2.5 Strategies and goals management

The Essential information model provides the concepts of *principles* and *objectives* for each of the layers in the information model, which enables for instance the definition of goals and strategies specifically for applications. The *strategy* class in the EAMTS 2008 information model was mapped to the *principle* class in the application layer of the Essential information model. Equally the *goals* were mapped to the concept of *objectives* in the application layer.

The *Essential Viewer* provides no predefined reports for viewing strategies and objectives, as well as their dependencies. For obtaining a suitable visualization the Microsoft Visio export extension may be leveraged. Since the conceptual view of the Essential information is rarely filled with reference data due to a lack of equivalent concepts in the EAMTS 2008 information model, the *allConceptual()* function of the export extension was used. The superfluous elements contained in the conceptual abstraction view were deleted manually and subsequently the layout of the symbols was adjusted resulting in the visualization shown in Figure 19.

For supplying information on the level of fulfillment concerning a goal, it is necessary to introduce the user defined attribute *reached* for the *goal* (objective) class of type float. Thereby, the user can evaluate all goals with respect to their fulfillment, for

instance by creating a query in the queries tab, which selects all goals with a *reached* value being greater or less than a certain value.

The *goal* (objective) class does not feature relationships to the *demand* class and the *project* class as it is suggested by the EAMTS 2008 information model. Therefore, those two relationships are added in order to be able to show the impacts of strategies and goals on demands or projects and thus also on application systems. An according impact analysis may be conducted manually by traversing from strategies via goals and demands/projects to application systems. Alternatively, custom tabular reports for this purpose may be defined as they are more convenient than traversing manually in the *Essential Modeler*. The swim lane diagrams, which are proposed by the scenario of strategies and goals management, cannot be created by the *Essential Viewer*.

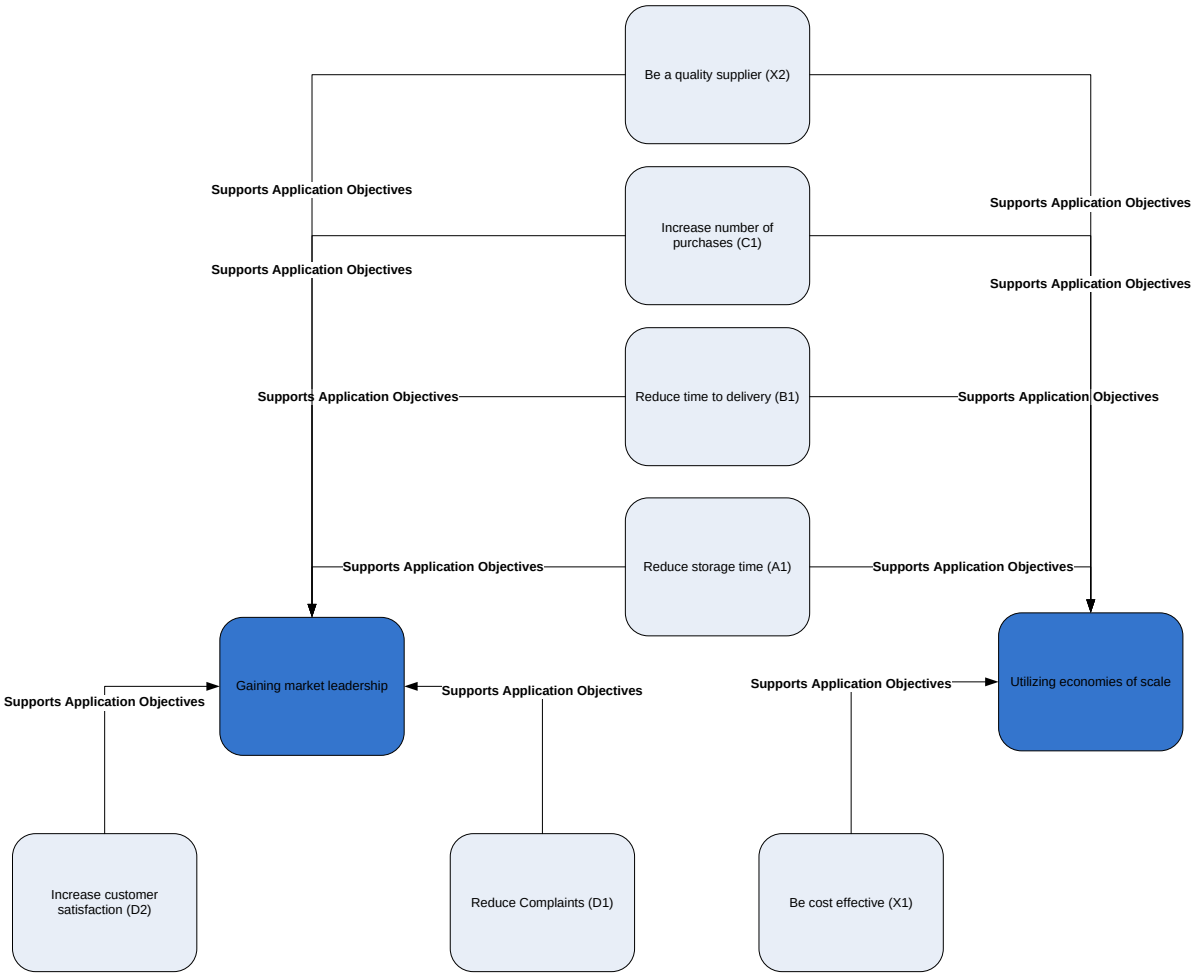


Figure 19: *Essential Project* – diagram containing strategies with their related goals

3.3.2.6 Business object management

The information model of *Essential Project* provides only little support for business object management. It contains the class *information representation*, which is suitable for representing business objects. The *information representation* class is associated with the class *application function implementation* that can be used for modeling the relationships between application systems and business objects. Furthermore, the user may specify one or more modes of the four offered interaction modes of *create*, *read*, *update*, and *delete* (see Figure 20).

There are no predefined reports shipped with the *Essential Viewer* for the topic of business object management. As a result, the user has to define custom reports in order to be able to view which application systems use which business objects. Alternatively, the user might use the *Essential Modeler* to explore the relationships between business objects and applications manually.

The Essential information model is not designed to support the modeling of interfaces. Instead, a *static application provider architecture* is defined for each application system, which is used to model the dependencies of a certain application system to other applications. Therein it is not possible to associate the business objects that are exchanged in the course of the information flows between application systems. For this reason, the visualizations demanded by the scenario of business object management cannot be generated on the basis of the Essential information model.

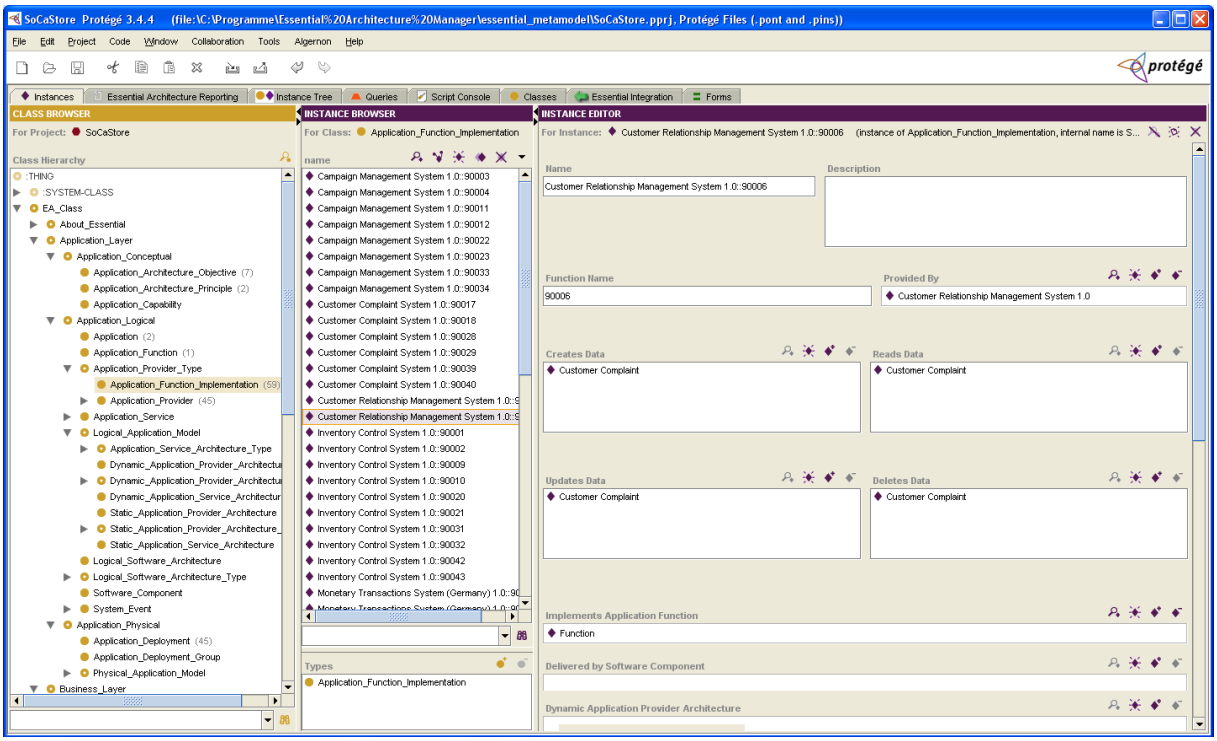


Figure 20: *Essential Project* – exemplary instance of application function implementation

3.3.2.7 SOA transformation

The SOA transformation scenario is supported in large parts by the *Essential Project*. This is mainly due to the flexibility of the information model provided by the adaption possibilities in the class tab. Hence, several attributes that are demanded by the scenario of SOA transformation were added to the already existing set of attributes in the *application system* (Essential: application provider) class. At first, a *domain count* attribute was introduced in order to maintain information on the number of different domains employing a specific application system. Furthermore, a *frequency of change* attribute was added indicating how often an application system is subject to change. For both attributes the allowed values of low, medium, and high were specified. A further adaption to the information model was made for the *business process* class. The scenario demands that a business process may be labeled as differentiating as opposed to standardized. A differentiating business process in this context is thus a business process that is critical for the company's unique selling position. This demand was realized by introducing the Boolean attribute *differentiating* and setting the default value for this attribute to false. In order to analyze the set of application systems and business processes with respect to the newly introduced attributes the capabilities of the queries tab may be leveraged (see Figure 21). Highlighting differentiating business processes is not supported by the Essential Viewer. Similarly, the color coding of application systems by their frequency of change in traffic light colors is not provided. Nevertheless, the additional attributes may be included as columns in customs reports that may be defined by the user for this scenario.

The Essential information model supports the modeling of services by the concept of *application services*. The services are designed to be provided by application systems and to support business processes. Moreover, there is a *status* attribute indicating whether the service is online or retired. However it is not possible to link the services to projects representing SOA transformation projects, which transform application systems into services. Thus, also the dependencies between transformations, which are given by the assigned projects, cannot be modeled with the Essential information model.

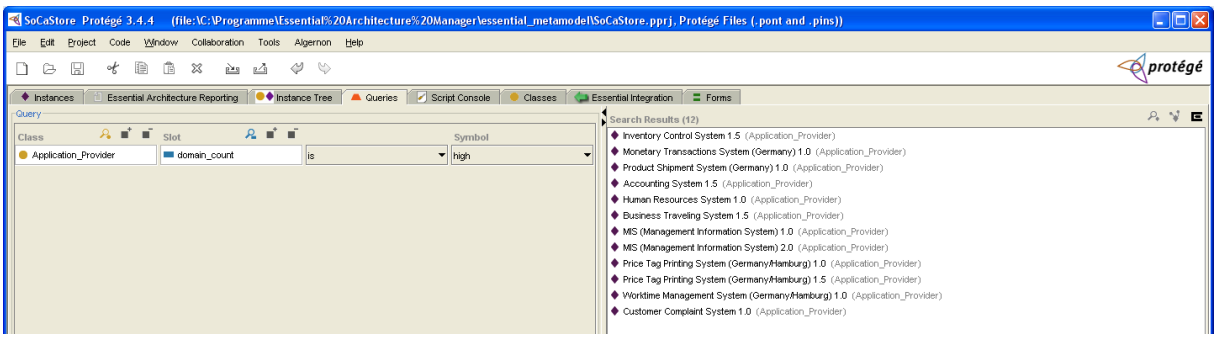


Figure 21: *Essential Project* – query for selecting applications with a high number of domains

3.3.2.8 IT architecture management

The Essential information model provides the necessary classes for supporting IT architecture management. The class *architectural solution* was mapped to the two classes of *technology product build* and *technology build architecture*. The first class is used for the relation to application system and the latter represents the *architectural solution* itself. The concept of *solution element* was mapped to the *technology product* class, which is prefilled in the *Essential Modeler* with a multitude of *solution element* instances. Via the *technology product role* class the *solution element* class is connected to the *blueprint element* class, which was mapped to the *technology component* class. In accordance with the *technology product* class, also the *technology component* and *technology product role* classes are prefilled with a variety of instances. Instances of the *technology product role* class describe for example that an Oracle 9i is used as a database. This class is finally connected via the *technology provider usage* class to the *architectural solution* class. Only for the class of *architectural blueprint* no suitable equivalent could be found in the Essential information model.

In order to evaluate the scenario of IT architecture management the introduction of two additional attributes became necessary. Firstly, the Boolean attribute *standard conformant* was added to the set of attributes of application system. In case an application does not have an assigned architectural solution the attribute value is set to false. The second attribute called *retirement*, which is also of type Boolean, was added to the *architectural solution* class and indicates that an architectural solution will be replaced at a certain point of time.

The queries tab in Protégé can subsequently be used to search for application systems conforming to the architectural standards. Furthermore, the queries tab enables the user to select architectural solutions that are marked for retirement. However, color-coding, which application system is compliant with an architectural solution, is not supported in the automatically generated cluster maps of the *Essential Viewer*. Visualizations that were created automatically on importing the reference data are provided by the *Essential Modeler*. Those visualizations indicate which solution elements are employed by a certain architectural solution. An exemplary visualization is shown in Figure 22.

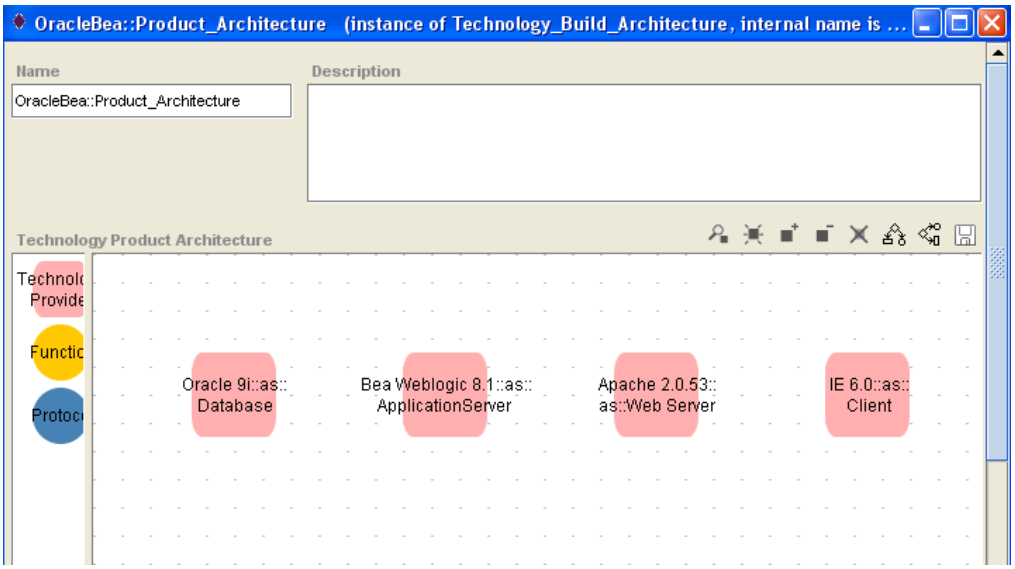


Figure 22: *Essential Project* - example visualization of the solution elements of an architectural solution

3.3.2.9 Infrastructure management

The Essential Project provides only limited support for the scenario of infrastructure management. The *infrastructure service* class in the EAMTS 2008 information model could not be mapped to a corresponding class in the Essential information model. Nevertheless, introducing several additional attributes and an additional relationship to the *solution element* class helped to accomplish a part of the infrastructure management scenario.

For every solution element it is required to have information on the specific organizational units or locations at which the element is hosted. For this purpose an additional relationship was added to the *solution element* class representing the association to organizational units (mapped to the *site* class in the Essential information model). By employing the queries tab the user can find out, which solution elements are hosted at a specific organizational unit.

Besides the relationship several attributes were added to the attribute set of the *solution element* class. Of special interest in the context of infrastructure management is the date at which the support for a certain solution element is terminated. For this reason the *support end date* attribute was introduced. However, it could only be modeled as an attribute of type string as a date attribute type is not supported by the *Essential Modeler*. A further important information for infrastructure management is the current lifecycle phase of a solution element. For the evaluation of the scenario the attribute values of this attribute were restricted to *introduction*, *production*, and *retirement*. Further attributes were introduced for the financial aspects of solution elements, as it is necessary to have information on the operating and licensing costs of a certain solution element.

The association of solution elements to application systems cannot be realized as demanded by the scenario due to the missing class for *infrastructure service*. Instead the solution elements of an application system can only be found out by traversing via the *architectural solution* class to the *solution element* class. However, this can only be achieved for application systems that conform to an architectural solution. For those which are non-conformant with any of the architectural solution, no relation to solution elements may be traced.

A further drawback in the context of the scenario of infrastructure management is that no predefined reports exist as all the attributes and the relationship necessary for infrastructure management were added to the information model shipped with the *Essential Project*.

3.4 Executive summary

The Essential Project pursues a meta-model based approach (see Section 2.2) to be configurable to the requirements of the users. The predefined information model, which is captured in Protégé, is organized in different layers, e.g. business, application, information, and technology architecture. In addition, the abstraction views of conceptual, logical, and physical view structure the information model. The information model can be extended and enhanced via the convenient class tab in Protégé.

Although the Essential Project does not include many predefined reports, it can be easily extended by custom reports and thus adjusted to the users' requirements. A new custom report is created by defining an appropriate xsl file and adding the report to the respective report classification and report group. Regarding visualizations, Essential Project only supports the types of cluster maps and graph layout maps. Furthermore, these types are out-of-the-box merely applied for a small number of visualization purposes. Nevertheless, Essential Project can be customized to support the other visualization types if the according adaptations to the predefined information model are made.

A deficiency of Essential Project is the absence of the time-related aspects of an EA. Protégé, for instance, does not provide a date data type and thus attributes, such as the "startRetirement" date of an application system, have to be modeled as string attributes. For analysis and visualization purposes the string attributes can be converted to data type date, however, this assumes that the string values for an actual date attribute have been recorded uniformly in an appropriate format throughout the entire repository. In the context of time-related aspects it must be further mentioned that Essential Project's information model has been designed to capture only the current state of the EA. Therefore, several concepts showing the evolution of an application landscape are not part of the predefined information model. For instance, the impacts of projects on the set of application systems at a certain point of time cannot be modeled without the according adaptations to the information model.

The Essential Project does not see itself as a tool that is meant to cover all EA management aspects at once. Instead, the user is encouraged to customize the tool in

order to meet the user's specific requirements. Having performed a customization to the tool, an everyday user may perform the various tasks associated to EA management via the intuitive and highly adaptable user interface.

4 Comparing open source tools for EA management

After providing a detailed overview on the evaluation of Essential Project, this chapter contains an introduction to two further open source EA management tools, which are similarly described according to scenarios of the EAMTS 2008. In the third Section 4.3 the three open source EA management tools are compared. Thereby, only the scenarios for analyzing specific functionality are considered. This is due to the characteristic of a flexible and configurable information model provided by two of the tools: Essential Project and Tricia/SyCaTool. Those two EA management tools pursue a meta-model driven approach, which is described in Section 2.2. Iteraplan in contrast has a predefined information model, which can only be adapted to a small extent. Therefore, the latter open source EA management tool merely provides support for scenarios for EA management related tasks that are supported by the information model. The other two tools on the contrary can be configured to meet the requirements of various scenarios. For this reason the scenarios for analyzing EA management support are not regarded in the comparison as well as in the following descriptions of the two additional open source EA management tools.

4.1 Iteraplan

Iteraplan is an EA management tool, which is offered in two versions by iteratec GmbH[9]. The open source community edition was released in 2008 and is available under the AGPL 3.0 open source license. The closed source alternative named enterprise edition, which of course offers significantly more features than the community edition 2.7, is not considered in the course of this comparison.

For the scenario of *importing, editing, and validating model data* only little support is provided by the Iteraplan community edition due to the missing import feature. Thus, importing Excel spreadsheets or XML files, for instance, is not possible. A solution to overcome this deficiency is to import the data directly via the database interface. Nevertheless, the Iteraplan community edition contains a number of predefined consistency checks in order to validate the consistency of model data. These consistency checks also serve as means for detecting missing values for mandatory attributes. Links to external resources and documentation may be added to any information object in the repository by specifying URLs or paths to the according files. Usually each information object is edited individually in the Iteraplan community edition, but there is also a feature for bulk updates or bulk deletes. This enables the user to modify properties, relationships, and attribute value assignments for several information objects of the same type simultaneously.

[9] http://iteraplan.de/

The scenario *creating visualizations of the application landscape* is well supported by Iteraplan. Almost all visualization types demanded by the scenario can be generated automatically by the tool. The visualization types, which are supported, include cluster maps, process support maps (Iteraplan terminology: landscape diagram), time interval maps (Iteraplan terminology: masterplan diagram), and portfolio matrices. The only visualization type not supported is the swim lane diagram. The diagrams can be generated in SVG, JPG, PDF, PNG, and Microsoft Visio format. On initiating the generation of a visualization the user may specify different characteristics. Depending on the type of visualization, for example, the classes included in the diagram or properties of the queried information objects have to be chosen. Furthermore, the settings for creating visualizations may be saved in order to enable reproduction.

Most of the visualizations generated by Iteraplan are not designed to be adapted manually by the user due to the output formats. Only diagrams generated to Microsoft Visio may be altered, but it must be kept in mind that those changes are not taken into consideration during regenerating the visualization. Highlighting objects of interest can only be done manually in Microsoft Visio diagrams. As the visualizations are decoupled from the underlying data, there are no adaptations that alter the visualization's semantics. Moreover, traversing from a specific element in the diagram to connected elements cannot be achieved by Iteraplan, although this is an important functionality in the *interacting with and editing of visualizations of the application landscape* scenario. Nonetheless, extensive filtering mechanisms are provided, which enable the selection of specific information objects to be represented in the visualization. Sorting and grouping according to one or more attributes in tabular reports is not offered directly by the tool. However, this deficiency can be overcome by exporting the query results to Excel and using the built-in functionalities for sorting and grouping.

The scenario of *annotating visualizations with certain aspects* investigates the ability of Iteraplan in visualizing certain attributes, e.g. by changing the background color of an element. Iteraplan supports color-coding in most of its visualizations and the color palette can even be specified by the user. In addition to color-coding, Iteraplan also provides the capability to visualize attributes by specifying different border types. Hence, two different attributes can be visualized together in a single diagram. Iteraplan does not support the visualization of certain attributes by adding thresholds, such as traffic lights. The tabular report showing certain attributes for application systems as demanded by the scenario, can be created by leveraging the Iteraplan report feature. Instead of a simple listing of the report contents, the results may also be exported to formats like Microsoft Excel, XML, CSV etc.

The web access to information as demanded by the scenario of *supporting lightweight access* is granted by the Iteraplan community edition. This is due to the fact that Iteraplan constitutes a web-based EA management tool, which encompasses both editing and viewing information via a web browser. A drawback in the context of this scenario is that the community edition does not provide user rights management[10].

[10] Only one user can be specified in the community edition.

Consequently the access of a user to certain visualizations or reports cannot be limited and furthermore, a specific user cannot be restricted from editing certain data. Visualizations in Iteraplan are created independently of each other and therefore navigation between related visualizations or reports is not possible. Again, the filtering feature as described in the previous scenario of *interacting with and editing of visualizations of the application landscape* can be leveraged to exclude certain objects from visualizations or reports. Searching for objects in the tabular reports can be realized by the browser built-in search. The search in visualizations depends on the format, i.e. in Microsoft Visio the built-in search can be used whereas in the other formats no search is provided. Nevertheless, Iteraplan offers useful search functionality for searching information objects directly. Thereby, a full text search as well as searching for certain characteristics of information objects is provided.

The scenario of *editing model data using an external editor* is not supported by the Iteraplan community edition. The XMI export is disabled in the community edition, however the reporting functionality may be leveraged to export the required data in a desired output format, such as Microsoft Excel, XML, CSV etc. The Iteraplan community edition provides no functionality for reimporting data and consequently the scenario is not supported.

The *adapting the information model* scenario is only supported in parts by Iteraplan. This is mainly due the fixed information model, which does not allow to add, adapt, or delete classes. The same is true for relationships between the classes. Despite that, the Iteraplan information model allows to add, adapt, or delete attributes for any class. An attribute has a certain attribute type (e.g. enumeration, date, etc.) and can be labeled as mandatory. Default values as perceived by the EAMTS 2008 are not supported. Instead Iteraplan provides functionality for setting standard values by using bulk updates. Thereby, it is important to notice that standard value is set during the bulk update and not on creating a new instance. Furthermore, it must be taken care that already edited values are not overridden by the update. In case an attribute with already assigned values is to be deleted, there is a notification and the deletion has to be affirmed. Already created attributes may be edited to the extent that the name, attribute group or mandatory label can be changed. The attribute type, however, may not be altered.

The community edition of Iteraplan is not designed for *handling large scale application landscapes*. Due to the missing import functionality the application landscapes consisting of up to 10 000 application systems could not be imported automatically. Furthermore, there is a limitation on the number of elements in a visualization, which is set to 50 in the community edition. Nonetheless, the report functionality shows no such limitation and may therefore leveraged to create respective reports. Iteraplan's bulk update feature constitutes a valuable asset for editing large amounts of data. In addition, the extensive search functionalities can be used for finding elements.

For the scenario of *supporting multiple users and collaborative work* it is of focal interest that a user's access to certain data or visualizations may be limited. The Iteraplan community edition does not support user rights management. Therefore, the community user has read and write permissions on any element in Iteraplan. Additionally, neither workflow management nor notification mechanisms in case of

certain events are supported. Despite that, Iteraplan supports multiple users by indicating unsaved elements, which are edited by multiple users, in red font. In case two users are editing the same element, the one who saves later ends up with an error message saying that the changes could not be saved. Therefore, the alterations of the seconds user have to be redone.

As far as *usability* is concerned, the Iteraplan user interface can be considered to be very intuitive and well-structured. An extensive user guide is provided by Iteraplan as well as an installation guide. Moreover, there is an Iteraplan community dashboard, which serves as a platform for contributing and for communicating with other community members.

4.2 Tricia/SyCaTool

Tricia is an open source Enterprise 2.0 platform, which provides integrated web collaboration services. It offers a variety of functionality such as wiki collaboration, personal and team blogging, file and directory sharing. In the context of EA management the hybrid wiki, which constitutes a lightweight semantic extension to Tricia, is of special interest. This hybrid wiki approach enables the users to collaboratively create an ontology or data model in a bottom-up fashion by adding structured content to any wiki page in the form of arbitrary named attributes or tags. Visualizations may be created by using the open source software- and system cartography tool (SyCaTool) extension of Tricia (see Figure 23).

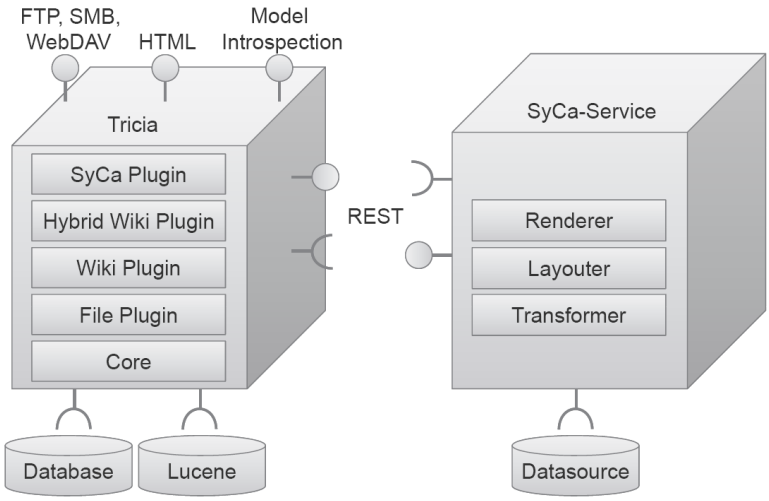


Figure 23: HybridWiki & SyCaTool architecture. Source: http://wwwmatthes.in.tum.de/wikis/sebis/hybrid-wiki, accessed 10.11.2010

The scenario of *importing, editing, and validating model data* is only supported in parts by Tricia/SyCaTool. Tricia offers no importing functionality for the structured content of the hybrid wiki pages and therefore the Excel spreadsheet import demanded by the scenario cannot be accomplished. For editing EA data in a convenient way, Tricia provides several features such as wiki pages that are organized hierarchically. Furthermore, it is possible to create internal links with auto-completion and link

checking. Consistency checks for validating the consistency of the EA data are not supplied. Nevertheless, broken internal links are highlighted immediately and active support for correcting them globally is offered. Due to Tricia's focus on web-based collaboration, a variety of external documentation may be linked to the EA content. Besides attaching documents, it is also possible to embed media files such as images, audio, video, and animation.

For the *creating visualizations of the application landscape* scenario support is provided by the SyCaTool. Cluster maps, time interval maps, and process support maps can be automatically generated by the SyCaTool whereas graph layout maps, portfolio matrices, and swim lane diagrams are currently not supported. Extensions to the predefined set of visualizations can only be created by writing Java code. The generated visualizations can be viewed and edited with the internal SyCaTool editor or may be exported to JPG, PDF, PNG, and SVG format. Depending on the type of visualization the visualization characteristics may be adapted. In the case of cluster maps, the outer and inner classes can be chosen by the user prior to generating the visualization.

The visualizations generated automatically by the SyCaTool are not meant to be adapted manually. Furthermore, no filtering mechanisms are provided in order to include only a subset of the application systems in a visualization. Similarly, the SyCaTool does not offer any functionality for traversing from a specific element to related elements. Therefore, the scenario of *interacting with and editing of visualizations of the application landscape* is poorly supported by Tricia/SyCaTool.

For the scenario of *annotating visualizations with certain aspects* it is important to what extent the SyCaTool supports the visualization of specific aspects, such as the maintenance costs of application systems. The attribute "availability" of application systems, for instance, can be visualized by changing the background color of the symbolic representations. Visualizing certain aspects by changing the border of symbols or adding thresholds like traffic lights is not offered by the SyCaTool. Hence, also the visualization of two aspects together in a single visualization is not possible.

The scenario of *supporting lightweight access* is well realized in Tricia as it provides a completely web-based approach to editing and visualizing EA information. Due to the extensive capabilities of Tricia in web-based collaboration, there is an elaborate user rights management. Access rights can be granted on several levels of granularity for read-only and read/write access. As access rights may be granted on the level of objects, the content of a visualization depends on the rights of the user generating it. The visualizations generated can be zoomed and the SyCaTool enables hiding or adding certain layers of the visualization. Moreover, there are no navigation mechanisms in the visualizations to get to more detailed diagrams or tabular listings of information. Nevertheless, extensive search functionality is provided, which includes full-text search over all content items and attributes as well as structured searches by content type or content space. However, this search functionality does not cover the automatically generated visualizations.

The *editing model data using an external editor* scenario is not supported by Tricia/SyCaTool due to the missing importing and exporting functionality.

As Tricia offers an information model, which can be completely configured by the users, the scenario of *adapting the information model* is supported. New classes for the information model can easily be created by introducing a new type tags. Similarly, classes may be deleted by deleting the desired type tags. Thereby, it must be kept in mind that the wiki pages formerly tagged with the respective type tag still exist. The Tricia/SyCaTool information model offers similar flexibility for attributes and relationships. New attributes or relationships are added by creating a new name-value-pair in the structured content section of a wiki page. At first, changing the attribute or relationship only impacts the instance represented by the wiki page on which the adaption was made. The other wiki pages with the same type tags receive propositions for the changed attributes or relationships. Deleting attributes or relationships and setting default values is not supported by Tricia. Furthermore, no mandatory attributes may be defined and setting cardinality constraints for relationship ends is not provided.

The scenario of *handling large scale application landscapes* is at first concerned with importing a large amount of data. As Tricia offers no importing functionality, this import cannot be achieved. Additionally, loading overview wiki pages and creating visualizations takes a long time due to the considerable number of objects in a large scale application landscape. Usually each instance, i.e. each wiki page is edited individually in Tricia. For large scale data, however, the search functionality may be leveraged to obtain a more convenient way of editing data. A dedicated search has to be defined and embedded in a wiki page. This results in a tabular view of the searched structured content, which features in-place editing. By defining a search on application systems with the according attributes and also embedding it in a wiki page, a report as demanded by the scenario may be generated.

The *supporting multiple users and collaborative work* scenario is supported by Tricia due to the fact that it has been designed as a collaboration platform. Tricia allows role based access on various levels of granularity reaching from overall access down to access to single information objects. Hence, information objects represented in a visualization are chosen based on the rights of the user generating the visualization. Furthermore, Tricia provides extensive functionality to save metadata, such as last modification date and last modifying user to enable traceability. These metadata attributes may also be used in the Tricia search functionality. Additionally, Tricia allows comparing different wiki page versions side-by-side and any previous version of a wiki page can be restored. A further useful feature in the context of collaborative work is blogging. Tricia offers a broad functionality for personal and team blogging, which may be used to discuss certain topics. Thereby, blog entries can be linked to certain wiki pages, which correspond to information objects, and vice versa. Tricia offers no workflow support and there are no notification mechanisms, which remind the users to update aged data. Nevertheless, the group email functionality may be leveraged to make up this deficiency.

Regarding *usability*, Tricia/SyCaTool can be described as intuitive with respect to the user interface and handling. Furthermore, an extensive and comprehensive developer documentation is provided as well as a video on working with the hybrid wiki. Nevertheless, a documentation concerning the operation of Tricia/SyCaTool in the context of EA management has not yet been supplied.

4.3 Comparison results

The three open source EA management tools described in the previous sections of the thesis were qualitatively compared to each other in the course of an expert group discussion. Two students, who had evaluated Essential Project and Iteraplan and three members of the sebis staff took part in the discussion. One of the sebis staff members is assigned to the Tricia research project and was therefore able to provide the required information. For each scenario analyzing specific functionality the abilities of the three tools were presented separately by the respective person and subsequently the meeting participants agreed on a ranking of the three open source EA management tools per scenario.

For the scenario of *importing, editing, and validating model data* Essential Project reached the first rank and both other tools came second. This is mainly due to the missing importing functionalities of Iteraplan and Tricia/SyCaTool whereas Essential Project provides a very flexible way of importing model data by its XML/XSL-based approach. The absence of values for mandatory attributes can only be highlighted with Essential Project and Iteraplan. Editing model data is facilitated in a similarly convenient manner by all three tools. External documentation may be linked to dedicated information objects by any of the investigated EA management tools.

Iteraplan offers the best support for the scenario of *creating visualizations of the application landscape* of the three open source EA management tools as it supports all visualization types demanded by the scenario except for swim lane diagrams. Furthermore, specific characteristics of the visualizations may be configured depending on the type of visualization. Tricia/SyCaTool, which reached the second rank, offers the automatic generation of cluster, time interval, and process support maps. Essential Project provides the weakest support regarding the creation of visualizations. Only graph layout maps and cluster maps can be automatically generated.

Essential Project and Iteraplan were ranked as the two best performing EA management tools for the *interacting with and editing of visualizations of the application landscape* scenario. Nevertheless all three tools did not offer a broad support for this scenario. In Iteraplan only visualizations in Microsoft Visio may be edited, but as those are decoupled from the information base, the adaptations made have no impact on the underlying data. Depending on the type of visualization, Essential Project offers editing visualizations with according alterations to the visualization semantics. In Essential Projects all visualizations except for the cluster maps published to the Essential Viewer can be edited. Furthermore, Essential Project offers navigation mechanisms both in the Essential Viewer and in Protégé. This traversing from a selected element to its related elements is not provided by Iteraplan. Though, Iteraplan other than Essential Project provides advanced filtering mechanisms for selecting information objects for reports or visualizations. As Tricia/SyCaTool does not offer any of the functionalities demanded by the scenario, it was rated last.

For the scenario of *annotating visualizations with certain aspects* Iteraplan offers by far the best support. All functionalities demanded by this scenario except for adding

thresholds such as traffic lights to visualizations, are supported by Iteraplan. However, adding traffic lights depending on the value of an attribute like availability of application systems, is not facilitated by any of the investigated EA management tools. Tricia/SyCaTool was rated second as it offers the possibility to visualize specific aspects such as maintenance costs of application systems by changing the background color of the respective symbol. Changing border types or adding thresholds depending on the values of a certain attribute is not supported. Essential Project offers the weakest support for this scenario as only the tabular report showing the specific aspects of each application system can be created.

Tricia/SyCaTool offers the broadest support for the scenario of *supporting lightweight access* followed by Iteraplan. Tricia as a web collaboration platform and also Iteraplan are completely web-based and therefore both viewing and editing model data is achieved via web access. Tricia's elaborate user rights managements allows to limit the access of a user on various levels down to certain information objects while Iteraplan does not provide user rights management in its community edition. Both tools do not support navigation in visualizations. Iteraplan offers extensive filtering possibilities for selecting information objects and their attributes for visualizations and reports whereas Tricia does not provide such functionality. All three EA management tools facilitate searching visualizations and reports and none of them offers the possibility to show or hide layers in visualizations. Unlike the other two tools Essential Project offers navigation in visualizations and reports in order to get from an overview diagram or item in a report to a detailed tabular listing of information of a specific information object. As Essential Project only supports viewing not editing information in a web based way and does not provided any of the other features demanded by this scenario, it was rated last.

In general the scenario of *editing model data using an external editor* is poorly supported by all three open source tools. Iteraplan and Tricia/SyCaTool do not support importing model data and therefore the requirements of this scenario cannot be fulfilled. Similarly, Essential Project is not designed to support editing model data in an external editor. Nevertheless, the user may configure appropriate exporting and importing procedures. But it is important to keep in mind that no check-out mechanisms are provided and thus the reimported data overrides repository data, which may have been edited in the meantime.

Essential Project provides extensive support for the scenario of *adapting the information model*. Classes, attributes, and relationships may be added, altered, and deleted. Furthermore, default values and mandatory attributes may be defined by the user. Additionally, Essential Project offers an exporting and reimporting functionality for its information model. Tricia/SyCaTool, which was rated second, supports creating and altering classes, attributes, and relationships. Deleting attributes and relationships is not possible, as well as setting default values or defining mandatory attributes. Iteraplan provides the weakest support for this scenario as it only allows adding, altering, and deleting attributes due to the fixed information model. Nevertheless, it supports defining mandatory attributes and the deficiency of no default values may be overcome by using the bulk update functionality. None of the three EA management tools provides a

visualization of the current information model and furthermore no symbolic representations may be defined for certain classes.

The scenario of *handling large scale application landscapes* could only be evaluated for Essential Project due to the missing importing functionalities of the other two tools. Therefore, the capabilities of Iteraplan and Tricia/SyCaTool could not be tested according to the scenario. For this reason, it was decided to refrain from rating the scenario of *handling large scale application landscapes*.

Tricia/SyCaTool and Essential Project supported the scenario of *supporting multiple users and collaborative work* best. Tricia with its elaborate user rights management may restrict the access of every single user on different levels of granularity. Essential Project provides less support for this evaluation item as the access to Protégé and the Essential Viewer is granted as a whole. Both tools save metadata on the last changes and offer no notification mechanisms in case of certain events. Nevertheless, both tools offer functionality for overcoming this deficiency. Tricia provides for instance blogging, which may be linked to certain information objects, and furthermore group emailing to inform a user group of certain events or changes. Due to its Protégé built-in collaboration capability, Essential Project offers adding comments to single information objects and live chat with other users. Iteraplan, which provides the weakest support for this scenario, has no possibility to limit the access rights to certain data or visualizations as there is no user rights management. Furthermore, no notification mechanisms or similar functionality is provided. All three open source tools have in common that they do not feature workflow management support.

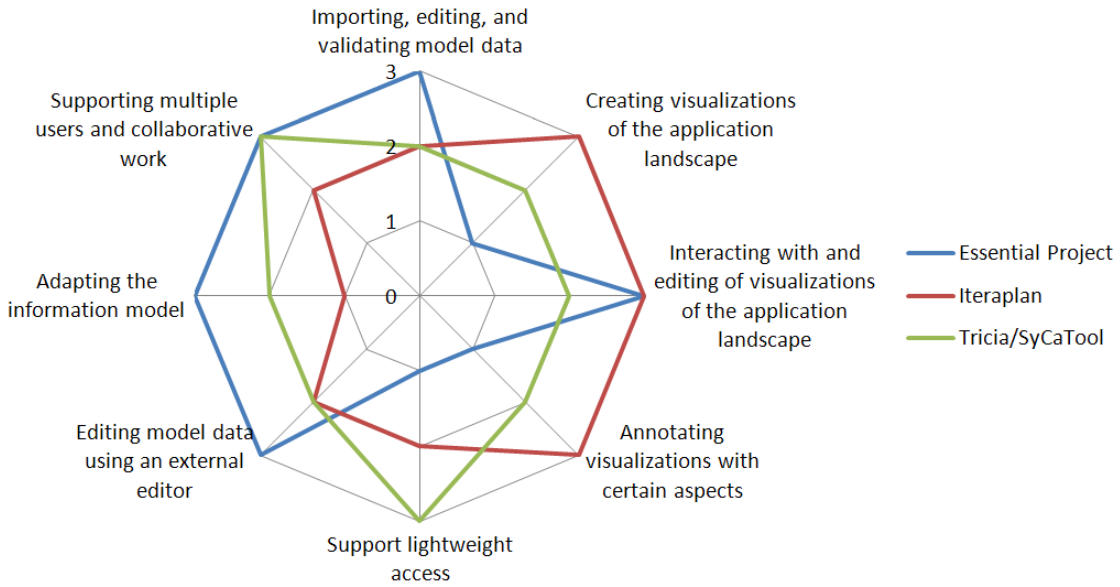


Figure 24: Kiviat diagram illustrating the evaluation results for specific tool functionality

Figure 24 shows a kiviat diagram illustrating the evaluation results for specific tool functionality. The axes of the diagram correspond to the eight scenarios for analyzing specific functionality, which were rated in the course of the group discussion. Thereby

the scale is negatively correlated with the rank that the tool achieved in the comparison, i.e. the EA management tool with the best support for a scenario is shown on the outer edge of the diagram and the ones with weaker support are located closer to the center. In the kiviat diagram the specific strengths and weaknesses of each of the three open source EA management tools become apparent. Iteraplan clearly shows its strengths in visualizing application landscapes whereas the support for alterations to the information model, importing or exporting model data, and collaborative work is rather poor compared to the other tools. In contrast Essential Project shows its strengths in the latter fields, which are information model centric. This supports the fact that the Essential Project started off with its Essential Meta-Model. The deficiencies of Essential Project clearly lie in its weak out-of-the-box support for visualizations. Tricia/SyCaTool with Tricia being a web collaboration platform shows its strengths in collaborative work as well as in the web-based user interface and provides a medium support for the other fields.

5 Enterprise-specific evaluation guide

This chapter of the thesis commences with depicting software evaluation processes in scientific literature. In particular, three software evaluation processes, which are later on used for compiling the evaluation guide, are delineated in detail in Section 5.1. The subsequent Section 0 is describes an EA management tool evaluation process that took place at a German communication service provider. Based on the results of the previous two sections, Section 5.3 presents an enterprise-specific evaluation guide for EA management tools.

5.1 Software evaluation processes in scientific literature

The topic of software evaluation received increasing attention during the late 90ies and the beginning of the new millennium. The rising costs for individual software production and software maintenance as a consequence of the increasing complexity and demand for user friendly software, forced companies to search for appropriate software products in the market. Hence, software evaluation received increasing attention and suitable methods and processes for software evaluation were frequently discussed in scientific literature [MT97, MN98, Oc00, Co04]. Especially in the field of commercials off-the-shelf (COTS) products a variety of contributions can be found.

There is a number of characteristics inherent in software evaluation, which have to be kept in mind during evaluating software products. Software evaluation may have different points of view depending on the actors involved in the process. Therefore, subjectivity is always present. Furthermore, the final decision usually depends on multiple criteria meaning that a number of attributes has to be selected in order to perform an evaluation. This selection is an important part and is dependent on the points of views under which the evaluation is executed. In addition, some of the attributes cannot be exactly defined and formalized. Furthermore, their relative importance as well as the according measurements are difficult to determine [Vl99]. Consequently, uncertainty also plays a role in software evaluation.

According to [Du94] software evaluation may be subdivided into a *model dependent evaluation* and a *model independent evaluation* (see Figure 25). The model dependent evaluation is based on the excerpt of reality, in which the software is to be employed. Thereby, in particular the functional aspects of the investigated software are explored to detect which specific tasks are supported. The model dependent evaluation may further be subdivided into quantitative and qualitative evaluation. Quantitative evaluation investigates the functional scope of the software whereas the qualitative evaluation explores the way in and extent to which the software supports the required tasks. The model independent software evaluation is concerned with evaluating the user interface and the evaluation of basic conditions, such as hardware and software requirements.

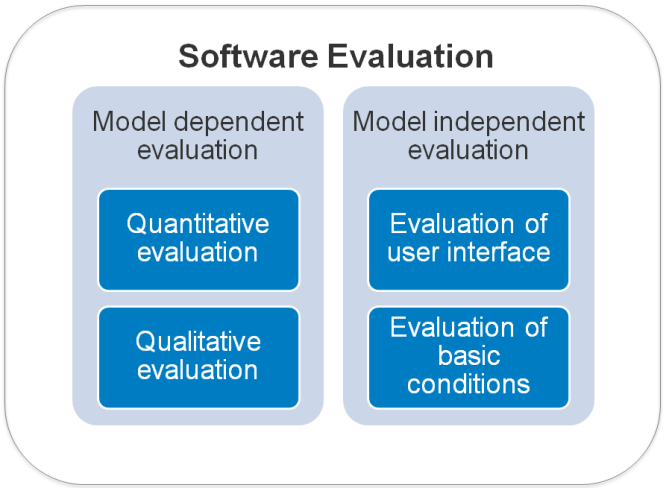


Figure 25: Software evaluation according to [Du94]

In the context of COTS products a variety of different evaluation methods and processes has been developed. These are relevant for compiling an evaluation guide for EA management tools due to the fact that an EA management tool can be considered as a COTS product. Some of the COTS evaluation and selection processes, however, are restricted to COTS products that are used in software development and are therefore not applicable for the purposes of this thesis [Oc00, Bu02]. COTS selection processes are concerned with determining the fitness-of-use of COTS products in a certain context. Thus, the decision maker is provided with the necessary information to select the best COTS product from several competing alternatives [MRE07]. So far none of the developed methods and processes can be considered as the silver-bullet because due to the distinct contexts different approaches are more suitable than others. Nevertheless, most methods share some key steps which are described by [MRE07] as the general COTS selection process:

- "*Step 1*: Define the evaluation criteria based on stakeholders' requirements and constraints.
- *Step 2*: Search for COTS products.
- *Step 3*: Filter the search results based on a set of 'must-have' requirements. This results in defining a shortlist of most promising COTS candidates, which are to be evaluated in more detail.
- *Step 4*: Evaluate COTS candidates on the short list.
- *Step 5*: Analyze the evaluation data (i.e. the output of *Step 4*) and select the COTS product that has the best fitness with the criteria. Usually, decision making techniques […] are used for making the selection."

From the large number of processes and methods in the context of COTS products, three fairly different approaches were selected to be depicted in this thesis. They are described in chronological order and were chosen due to certain characteristics that are important to the enterprise-specific evaluation guide presented in Section 5.3. The enterprise-specific evaluation guide has been developed based on two prerequisites.

Firstly, the evaluation process is required to be organization-specific and secondly it has to reuse the results of the EAMTS 2008. Due to the latter prerequisite the enterprise-specific evaluation guide is designed in a scenario-based way. IusWare was one of the first approaches that formalized an evaluation and selection process. It provides concise definitions of the terms relevant to an evaluation process and separates clearly the design of the evaluation model from its application. IusWare's three top level activities of *problem formulation*, *design evaluation model*, and *apply evaluation model* were used as a starting point for defining the enterprise-specific evaluation guide, which was subsequently extended by incorporating elements from PORE, PECA, and the EA management tool evaluation process in practice presented in Section 5.2. PORE as the second approach described details on gathering information based on questionnaires and puts an emphasis on supplier-led demonstration sessions as well as the design of appropriate test cases for the demonstrations. Those test cases correspond to scenarios why the prerequisite of a scenario-based evaluation process is met by including this part of PORE in the enterprise-specific evaluation guide. PECA provides a flexible approach to software product evaluation and has been explicitly designed to be tailored to the particular needs of an organization. Thus, the second prerequisite of enterprise-specificity is fulfilled.

IusWare is a very formal methodology that evaluates software products based on the multi criteria decision aid approach[MT97]. According to the authors a clearly defined evaluation process is necessary as many actors with different objectives are involved and a lot of information is produced in the course of an evaluation process [MT97]. Therefore, they contributed IusWare, which is composed of the three top level activities of *problem formulation*, *design evaluation model*, and *apply evaluation model* (see Figure 26).

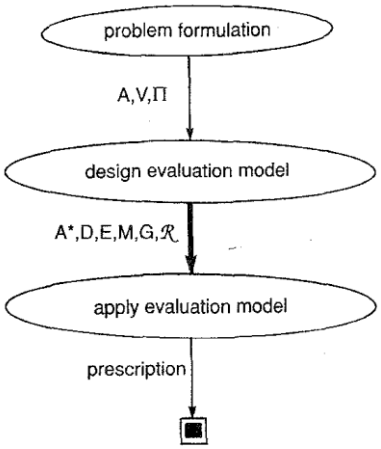


Figure 26: *IusWare* – top level activities of the evaluation process. Source: [MT97]

During the *problem formulation* activity a triple comprising the set of software products A, a set of points of views V, and a problem statement Π is defined. The set of software products describes the tools, which are to be investigated in the course of the evaluation process. The set of points of views depends on the actors that are involved in the evaluation. Thus, the bias of the evaluation is documented. The problem statement

defines what the outcome of the evaluation is expected to be. Thereby, it may have different values [MT97]:

- *Choice* leads to partition of the set of software products into a best products set and rest products set.
- *Sorting* results in a partition of the set of products according to previously defined profiles such as good, bad, etc.
- *Classification* ranks the set of software products from the most preferred to the least preferred.
- *Description* provides a formal description of the products without ranking.
- *Conceptualization* identifies ideal or quasi-ideal products, which are not available at the moment.

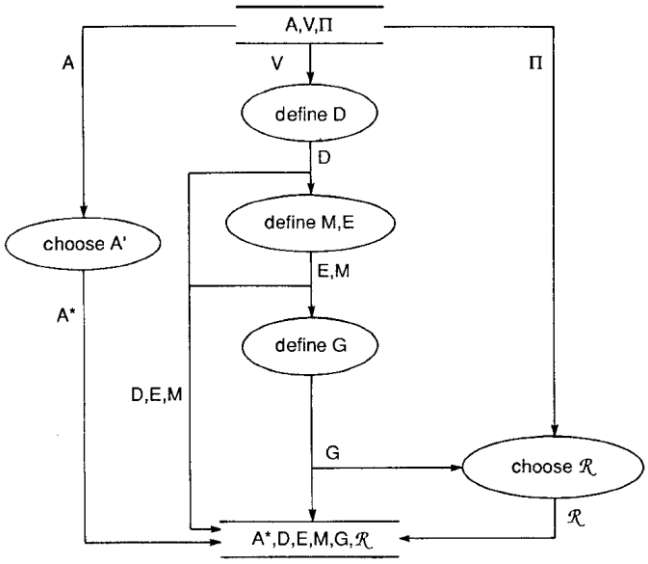


Figure 27: *IusWare* – design evaluation model activity. Source: [MT97]

The second activity *design evaluation model* is based on the outputs of the previous activity and contains three main streams of action: the choice of A^*, the definition of D, E, M, G, and the choice of R (see Figure 27). The set of software products A is purged to A^* in order to satisfy the independence constraint, to reduce the evaluation effort, and to meet any mandatory requirements. In the second stream the set of evaluation attributes D is defined by deriving it from the points of views in V. In a further step, the set of measures M and the set of scales E associated to the elements in D are defined. Scales may thereby be of any type including nominal. Subsequently, rules for transforming measures into preferences are defined (define G). In the third stream an appropriate aggregation technique R is chosen. According to [MT97] the aggregation technique mostly used in everyday evaluation is the weighted average sum. Nevertheless, the weighted average sum technique may incorporate a dangerous effect called compensation. This occurs, if incomparable attributes are defined, judged and the aggregated score leads to an indifference between products.

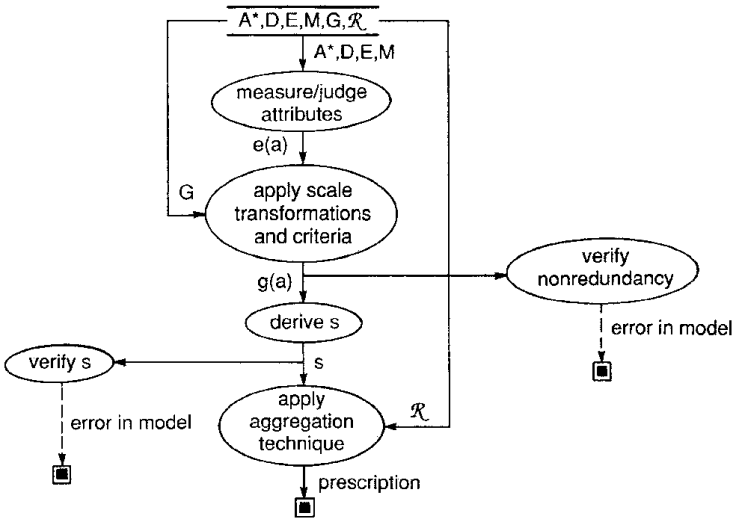


Figure 28: *IusWare* – apply evaluation model activity. Source: [MT97]

In the third activity of *apply evaluation model* the previously defined evaluation model is applied in order to obtain a prescription. In a first step, for each measured attribute and for each software product in A^* a measurement is performed whereas all non-measured attributes are judged. In the *apply scale transformations and criteria* step all nominal scales are transformed and the according numerical values are computed. Subsequently, the non-redundancy of basic and composed criteria is verified. This is an important step as in cases where there is redundancy and a weighted sum method is used, a particular product performing well in a redundant attribute is privileged in an unacceptable way. If no redundancy could be detected, the preference structure s is derived and verified. Thereby, it is tested whether the obtained preference structure models the client's preferences and needs. In the last step the selected aggregation technique is applied iteratively to the preference structure s, which leads to a prescription.

The PORE (Procurement-Oriented Requirements Engineering) approach, which was introduced in [MN98], focuses on the requirements engineering aspect of COTS product procurement. PORE suggests an iterative and parallel process of requirements acquisition and product evaluation as well as selection. This means that the number and detail of customer requirements is increasing at the same time as software products are rejected, which leads to a decreasing number of candidate products. Thereby, the requirements acquisition process enables an appropriate product selection and the product selection process in turn informs requirements acquisition. The iterative PORE process, shown in Figure 29, is executed until a COTS product is found that satisfies a sufficient number of the customer requirements.

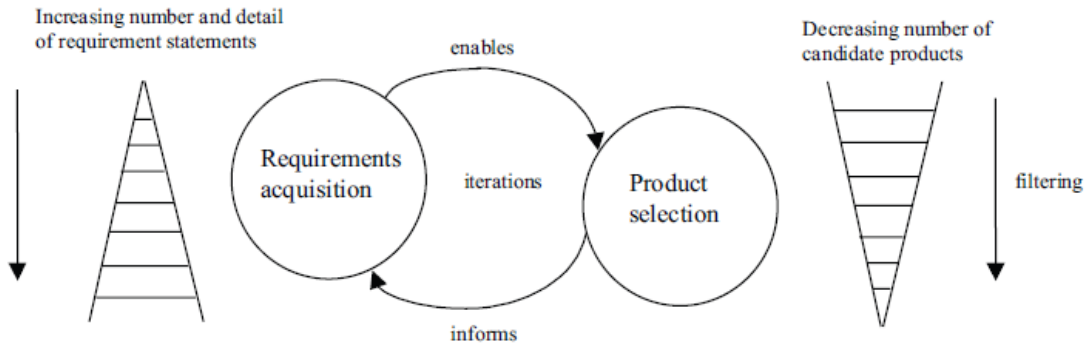


Figure 29: *PORE* – iterative process. Source: [NM99]

Besides the iterative process, PORE defines three templates that provide support for the three key stages of the process. *Template 1* is designed to be used during the early stages of the PORE process when the evaluation team has to rely on supplier data in sales brochures, technical documents, telephone conversations, responses to questionnaires, web site information, and market analyses [MN98]. Thereby, the purpose is to gather basic information about products and suppliers, technical product features, technical support arrangements (licenses, availability of source code, etc,), historical information about product and supplier, and essential functional user requirements. For this purpose template 1 suggests a number of activities. The basic product information is to be gathered by reading the product documentation. Furthermore, first-pass customer requirements have to be gathered by using simple techniques like brainstorming or interviewing. A questionnaire, which is to be sent to each supplier, has to be developed. The purpose of this questionnaire is to get basic supplier and product information as well as to check how much each of the software products complies with the initial user requirements. In a further step the questionnaire responses are evaluated and software products that do not comply with the essential customer requirements are rejected. Furthermore, the evaluators have to become familiar with the candidate products by working with the demonstration copies. More customer requirements may be discovered by exploring desirable properties not found during the previous requirements acquisition. This can be achieved by using techniques like structured interviews and prototype walkthroughs. A further activity is to acquire detailed customer requirements by using scenarios to design test cases for software product evaluation.

Template 2 provides guidance for acquiring requirements and rejecting candidate products based on supplier-led demonstrations [MN98]. Such supplier-led demonstrations are commonly held during product selection. As a thorough preparation of a demonstration session is essential, template 2 describes several activities for this purpose. At first, simple working prototypes have to be set up in order to discover further customer requirements and to improve test cases for product evaluation. Additionally, it must be taken care that stakeholder representatives are present during each demonstration session. Furthermore, it is recommended to work with stakeholders to weigh the collected customer requirements and finally decide on effective units of measurement for product-requirement compliance scores. During each demonstration session it is recommended to ask questions about the product. A further activity during this stage is the allocation of compliance scores only if the product properties are

demonstrated. Moreover, it is advised to record decisions behind compliance scores by using video or having an independent scribing record. After each supplier-led demonstration session more customer requirements may be acquired by using different forms of card sorting or laddering techniques. *Template 3* is not contained in the literature sources as it was not completed at the time of the publication. This template is concerned with acquiring customer requirements and product information from customer-led product exploration.

The PECA (Plan, Establish, Collect, Analyze) is a high level process for evaluating COTS products as it does not detail on techniques needed in the course of an evaluation. The PECA process has further been designed to be tailored by each organization to fit its specific needs. The decision for an actual product is not part of the process, because the PECA process has been designed to provide the necessary information for the decision making [Co04]. Figure 30 shows the elements of the PECA process, which are not always executed sequentially due to the fact that certain events, such as the discovery of inadequate data, redirect the process flow to the appropriate element.

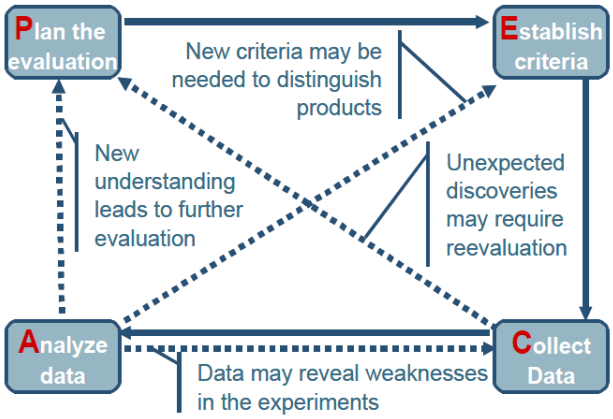


Figure 30: *PECA* – elements of the process. Source: [Co04]

The PECA process starts off with *planning* the evaluation. At first the evaluation team is formed by selecting team members such as technical experts, domain experts, contracts personnel, business analysts, security professionals, maintenance staff, and various end users. Thereby, a good balance of power between the team members is important and practical experience has shown that the number of members of the core working team should not exceed seven to eight individuals [Co04]. Next, a charter for defining the evaluation scope, goals, and constrains is created. Additionally, the charter includes the names and roles of all evaluation team members. In a further step the stakeholders of the evaluation are identified. Subsequently, the approach is picked by defining the depth and rigor of an evaluation and by choosing an appropriate selection strategy, which is either first fit or best fit depending on the objectives of the evaluation. The last step of the planning activity is concerned with estimating resources and the schedule. Only few techniques for estimating resources and schedules have been adjusted to fit the needs of COTS evaluation. Nevertheless, general techniques, such as expert opinion, analogy, decomposition, and cost modeling may be used for this purpose.

The activity of *establishing evaluation criteria* is at first concerned with identifying the evaluation requirements. In doing so, it must be taken care that not too many requirements are assigned to an evaluation as then suitable COTS products may be discarded because they do not meet all the requirements [Co04]. Moreover, errors may also occur, if the set of evaluation requirements is incomplete and therefore unsuitable products are chosen due to an insufficient understanding and oversimplification of the evaluation problem. In a further step the evaluation criteria are constructed by defining a capability statement and a quantification method. Well-defined criteria are thereby required to be discriminating and have to show minimal overlap.

The *collecting data* activity of the PECA process is concerned with executing the evaluation as planned previously in order to determine the performance of the investigated software products against the defined evaluation criteria. In the course of this activity different types of techniques may be used [Co04]. Data may be collected by reviewing literature such as user manuals, release notes, web based reports, product history, third party evaluations, etc. Furthermore, it is recommended to appraise the vendor by collecting information based on interviews, vendor literature, formal capability evaluations, independent financial analyses, trade journals, and customer kudos as well as complaints. [Co04] suggests to do hands-on experiments in order to investigate specific features of each software product. This can be achieved by conducting scenario-based evaluations, benchmarks, experimental fielding, and product demonstrations, in which the users have control.

The last activity of *analyzing results* comprises the consolidation of previously collected data. A consolidation always encompasses some loss of detailed information why it must be taken care that a there is a good balance between the need for easy understanding and the risk of losing too much information [Co04]. The consolidated data is subsequently analyzed in order to be able to make a recommendation. Several techniques like sensitivity analyses, gap analyses, and the analysis of the cost of repair have proven useful in this context [Co04]. Sensitivity analysis shows how the evaluation results react to changes in weighting of criteria, for instance. Gap analysis considers the product performance with respect to the evaluation criteria and thus shows how well a software product meets a criterion, or which functionality is missing. Analysis of the cost of repair assesses the work that must be devoted to make up the deficits of a product. The last step in the *analyzing results* activity is making recommendations. Thereby, three main outputs are produced. The product dossier is compiled for each product and contains software documentation, discovered facts, assessment results, etc. The evaluation record describes the evaluation itself and encompasses evaluation plans, personnel involved, dates and details of meetings, configurations, lessons learned, etc. Finally, the summary or recommendation document provides an outline of the evaluation and conveys this information to the decision-maker. It further includes the analysis of fitness and also describes evaluation deficiencies [Co04].

5.2 An EA management tool evaluation process in practice

After looking in detail at software evaluation processes in scientific literature, a practical perspective is taken by revisiting an EA management tool evaluation process, which was executed at a German communication service provider. The EA team of this company is rather small and is made up of one FTE that is split among some of the five IT architects and the head of IT architecture. The goals of the company's EA management approach are creating transparency, cost reductions, and improving strategic planning. Prior to selecting an EA management tool several tools were employed for achieving EA management tasks. Besides the Microsoft office tools of Excel, PowerPoint, and Visio a project tool of an IT consultancy was used.

The evaluation process was initiated and conducted by the head of IT architecture of the communication service provider. The evaluation was primarily IT driven, but management was involved and the final decision for the EA management tool was made on CIO and CTO level. In addition to the in-house capabilities, a consultant of a large IT consultancy supported the evaluation process. The evaluation approach was defined based on the experiences of one of the IT architects and the consultant. Thereby, no scientific literature besides the EAMTS 2008 was used. In total the evaluation process from initiation to decision for an EA management tool took three months.

A phase concerning problem formulation or defining goals for EA management tool deployment was not part of the evaluation approach. The evaluation process instead started off with a preselection phase, which was based on the results of the EAMTS 2008. In the course of this phase the scenarios for analyzing specific functionality and the scenarios for analyzing EA management support of the EAMTS 2008 were weighted according to their importance for the communication service provider. Furthermore, knock-out criteria were defined. The evaluation results of the investigated EA management tools in the EAMTS 2008 were analyzed with respect to the knock-out criteria and the weighting. As a result, a shortlist with three EA management tools was obtained.

The actual assessment of the remaining three tools in the shortlist was accomplished by involving the respective tool vendors. A questionnaire with questions on, for instance, the vendor, costs, licensing, software and hardware architecture was compiled and passed on to the tool vendors for completion. The answers to the questionnaire were weighted and rated by the evaluation team according to a defined scale. Based on the aggregated ratings, a ranking of the three EA management tools was compiled. In addition to the questionnaire, a number of custom scenarios, which represent the essential use cases of the EA management tool at the communication service provider, were defined. The evaluation team held separate in-house meetings with the vendors of the remaining three EA management tools. Thereby, each of the vendors was asked to demonstrate the custom scenarios without having had the opportunity to prepare their tool for the scenarios as they did not receive any information on the custom scenarios prior to the meetings. The capabilities of the tools were again rated and aggregated, resulting in total points for each of the three EA management tools. Together with cost estimations based on a total cost of ownership estimation approach [FP02], the

evaluation results were presented to the CIO and CTO, who finally approved the recommended EA management tool.

5.3 Compiling an enterprise-specific evaluation guide for EA management tools

The enterprise-specific evaluation guide for EA management tools, which is compiled in the course of this thesis, has to comply with two given prerequisites. Firstly, the evaluation process must be, as the name suggests, organization-specific. Therefore, the placement of EA management in a company's organizational and process structure must be respected as this influences the applicability of the EA management tool. Furthermore, a company's general goals of EA management have an impact on the way the EA management tool is used. Besides, enterprise-specificity also affects the resources that are allocated to an evaluation endeavor. The second prerequisite is that the evaluation guide has to be designed in a scenario-based way and reuses the results of the EAMTS 2008.

The enterprise-specific evaluation guide was compiled based on the EA management tool evaluation process in practice as well as the software evaluation processes in literature described in the previous sections. Thereby, the three top level activities of IusWare (problem formulation, design evaluation model, apply evaluation model) were taken as a starting point and complemented with further activities where necessary. In this step it was also decided that the tool selection itself should not be part of the evaluation process. Instead it terminates with a recommendation to the decision-maker. The activity of *defining the evaluation project* is in large parts based on the *planning the evaluation* activity of PECA. The *preselection* activity was taken from the EA management tool evaluation process in practice. Parts of the PORE approach are incorporated in identifying requirements in the *define evaluation* activity. Furthermore, PORE's recommendations concerning demonstration sessions were integrated into the *execute evaluation* activity.

The evaluation guide starts off with the *problem formulation* activity, which is concerned with describing the target state considering the EA management tool's utilization in the organizational and process structure of the enterprise. Moreover, the goals for the EA management tool are at first derived from the general EA management goals and subsequently prioritized.

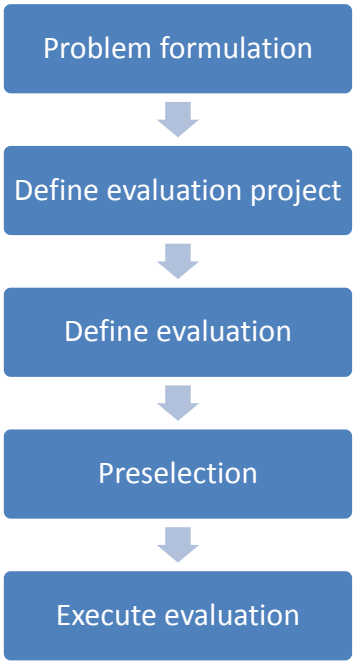


Figure 31: Top level activities of the enterprise-specific evaluation guide for EA management tools

The second activity of *defining the evaluation project* deals with setting the basic conditions for the evaluation endeavor. Firstly, the evaluation team is formed by examining the required range of skills to perform the EA management tool evaluation and selecting appropriate team members. Depending on the effort that is to be devoted to the evaluation the number of team members and the diversity of roles vary. Roles that could contribute to the evaluation of an EA management tool are, for instance, enterprise architects, IT architects, maintenance staff, representatives from departments, various end users, and the upper management. However, it must be kept in mind the core working team of an evaluation should be limited to approximately seven to eight individuals [Co04]. After the evaluation team members have been assigned to the evaluation project, their according roles and responsibilities within the project are defined. As a project usually has a defined start and end date, the time frame for the conduction of the evaluation must be agreed upon. Next, the resources needed for the evaluation of the EA management tools are estimated by relying techniques such as expert opinion and analogy [Co04]. A further important task in the *defining the evaluation project* activity is the identification of the evaluation stakeholders. A subset of the stakeholders is usually included in the evaluation team, but the remaining stakeholders are also needed for identifying requirements in later steps of the evaluation process. Subsequently, the longlist containing the EA management tools to be investigated in the course of the evaluation is compiled.

At the beginning of the *define evaluation* activity the evaluation requirements are derived from the EA management tool goals. Further evaluation requirements may be obtained by interviewing and holding brainstorming sessions with stakeholders or by studying documents on the EA management tools. The documents may be sales brochures, technical documents, user manuals, and web site information. The evaluation requirements from the various sources have to be consolidated and mapped to the

scenarios of the EAMTS 2008. Evaluation requirements that are not covered by the EAMTS 2008 scenarios can be captured by defining custom scenarios. Subsequently, the EAMTS 2008 scenarios and the custom scenarios are weighed according to their importance in the company. In a further step, a questionnaire is designed with the aim of obtaining basic product and supplier information as well as information on characteristics of the tools that cannot be detected by the scenarios. Next, appropriate scales that can be of any type (nominal, ordinal, etc.) are defined. In order to make judgments according to these scales summable, transformations into numerical values have to be defined. In case a formal approach is to be taken, an aggregation technique, such as the weighted average sum, has to be chosen. For a less formal approach, prioritizing may be considered to be sufficient. A last step in the *define evaluation* activity is the determination of knock-out criteria.

The activity succeeding the *define evaluation* is the *preselection* activity. During this step of the evaluation process the longlist of EA management tools is reduced by rejecting tools on the basis of the scenarios contained in the EAMTS 2008. Thereby, the knock-out criteria as well as the weighing of scenarios is taken into consideration when eliminating EA management tool candidates. The result of the *preselection* activity is a shortlist.

The last activity of the enterprise-specific evaluation guide for EA management is called *execute evaluation*. In the course of this activity each EA management tool vendor is provided with the previously defined questionnaire. It is important to set a deadline for replies in order to be able to start the evaluation of the questionnaire responses at a defined point of time. In a further step, selected EAMTS 2008 scenarios and all custom scenarios are simulated with attending stakeholders. The presence of the stakeholders ensures that judgments made for the scenarios are based on domain knowledge. The simulation of the scenarios may be achieved in two different ways. Firstly, a supplier-led demonstration session may be conducted. Secondly, the scenarios can be simulated in the course of EA management tool explorations led by the evaluation team. This is a way of achieving the simulation of the scenarios in cases where there is no tool vendor which is for instance the case for open source EA management tools. During the simulation of the scenarios the performance of each EA management tool is judged by each member of the evaluation team present at the demonstration or exploration session. Afterwards, the team members have to agree on a single judgment for each evaluated scenario of an EA management tool and the aggregation technique selected in an earlier activity is applied in order to obtain an overall score for each evaluated EA management tool. The output of the EA management tool evaluation process is an evaluation report, which contains a description of the evaluation itself, a recommendation for the decision maker, and an assessment of the evaluation deficiencies.

6 Recapitulation and prospects

This last chapter of the thesis provides a concise summary and reflection of the findings and concludes with an outlook on future research possibilities.

6.1 Recapitulation

The objectives of this thesis were the evaluation of the open source EA management tool Essential Project, its comparison to two other open source EA management tools as well as the development of an enterprise-specific evaluation guide for EA management tools. The first chapter of the thesis serves as an introduction to the thesis' motivation, objectives, environment, and its structure. The subsequent Chapter 2 commences with a depiction of the basics of EA management, followed by a detailed description of the EAMTS 2008 in Section 2.2. The last section presents the basic concepts of the open source movement and its influence in the context of EA management.

The evaluation of the open source EA management tool Essential Project was conducted according to the approach taken in the EAMTS 2008 (see Section 2.2) and is described in detail in Chapter 3. The evaluation yielded that Essential Project provides a meta-model driven and flexible approach to EA management. The flexibility became apparent in several different characteristics of Essential Project. Firstly, the importing and exporting functionalities are highly adaptable to the user's needs due to the XSL-based approach. Furthermore, the information model itself, although it is extensive and elaborate, can be adapted to organization-specific requirements. Thirdly, the flexibility of Essential Project can be seen in its adjustable and extensible set of visualizations and reports. Nonetheless, only few visualizations are supported out-of-the-box and moreover several visualizations demanded by the scenarios of the EAMTS 2008 could not be created due to the missing support for certain types of visualizations, as for instance process support maps. A further deficiency of Essential Project is that only modeling the current state of the EA is supported.

Chapter 4 presents the comparison of the three open source EA management tools of Iteraplan, Tricia/SyCaTool, and Essential Project. In the first two sections the capabilities of Iteraplan and Tricia/SyCaTool are delineated according to the scenarios for analyzing specific functionality of the EAMTS 2008. Section 4.3 describes the comparison results obtained through an expert group discussion. Thereby, the specific strengths and weaknesses of the three open source EA management tools became apparent. Essential Project shows its strengths in importing/exporting data and in its flexibility with respect to the information model and reports whereas its support of visualizing application landscapes is rather poor when compared to the other two tools. By contrast, Iteraplan shows its strengths in creating visualizations and provides weak support for importing and exporting model data, adaptations to the information model, and collaborative work. Tricia/SyCaTool offers a strong support for collaborative work and provides a medium support for the other investigated fields.

The compilation of an enterprise-specific evaluation guide is described in Chapter 5. In the first section processes found in scientific literature for evaluating software products

in general are delineated. The evaluation processes described in Section 5.1 were chosen based on the prerequisites of the evaluation guide to be developed. Based on an interview with a German communication service provider, a depiction of an EA management tool evaluation process in practice is provided in Section 5.2. Taking the results from the previous two sections into account, the enterprise-specific evaluation guide for EA management tools is compiled in Section 5.3.

6.2 Prospects

As the open source tools introduced in the course of this thesis are constantly evolving, it is recommended to redo the evaluation as well as the comparison of the three EA management tools after a certain period of time. Especially in the case of Essential Project a number of advances were detected during the evaluation period, in particular in the documentation as well as the share extensions web site. Furthermore, new open source EA management tools might be released. Therefore, a prospect might be to monitor the market for open source EA management tools and to contribute evaluations of new tools as well as to extend the comparison of open source EA management tools.

Open source EA management tools have so far only been compared to each other. Therefore, a comparison of the three open source EA management tools to proprietary EA management tools may be conducted. However, comparing the open source EA management tools to the high end proprietary tools would not be too beneficial as those contain a significantly broader range of functionality. Nevertheless, a comparison to tools at the lower end might prove useful as this can be taken as a basis for decisions on whether to invest in the adaptation of an open source tool or to use a proprietary EA management tool.

A further prospect of this thesis is to investigate the adaptability of the open source EA management tools in detail. Information on how cumbersome it is to, for example, add new or to adapt existing visualizations could provide a valuable decision basis during the selection of an EA management tool.

The enterprise-specific evaluation guide for EA management tools, which has been developed in the course of this thesis, has not yet been validated. The evaluation process might be reviewed by experts in order to identify improvement potential. Moreover, practitioners could apply the process in practice and thus discover its deficiencies.

A further prospect is to complement the enterprise-specific evaluation guide for EA management tools with detailed role descriptions for the different roles evaluation team members can adopt. In addition, a description of the various techniques that can be used in the course of an evaluation process might be compiled. A further useful instrument in the context of the evaluation guide is a template for the questionnaire that is sent to the EA management tool vendors.

A. List of abbreviations

EA Enterprise Architecture

FTE Full-time equivalent

IT Information Technology

B. Bibliography

[AB02] Asklund, U.; Bendix, L.: *A study of configuration management in open source software projects.* In: IEE Proceedings Vol. 149(1), pages 40-46, 2002.

[ARW08] Aier, S.; Riege, C.; Winter, R.: *Unternehmensarchitektur - Literaturüberblick und Stand der Praxis.* In: Wirtschaftsinformatik, Vol. 50(4), pages 292-304, 2008.

[BMS10] Buckl, S.; Matthes, F.; Schweda, C.M.: *Towards a Method Framework for Enterprise Architecture Management - A Literature Analysis from a Viable System Perspective.* In: 5th International Workshop on Business/IT-Alignment and Interoperability (BUSITAL 2010), Hammamet, 2010.

[Bu02] Burgues, X.; Estay, C.; Franch, X.; Pastor, J.A.; Quer, C.: *Combined Selection of COTS Components.* In: ICCBSS'02, Orlando, Florida, pages 86-96, 2002.

[Bu07] Buckl, S.; Ernst, A.M.; Lankes, J.; Schneider, K.; Schweda, C.M.: *A pattern-based approach for constructing enterprise architecture management information models.* In: Wirtschaftsinformatik, Karlsruhe, Germany, Universitätsverlag Karlsruhe, pages 145 - 162, 2007.

[Bu08a] Buckl, S.; Dierl, T.; Matthes, F.; Ramacher, R.; Schweda, C.M.: *Current and Future Tool Support for EA Management.* In: Steffens, U.; Addicks, J.S.; Streekmann, N.: MSS, SOA und IT-Management (MSI 2008), Workshop, Oldenburg, September 2008, GITO-Verlag, Berlin, 2008.

[Bu08b] Buckl, S.; Ernst, A.; Lankes, J.; Matthes, F.: *Enterprise Architecture Management Pattern Catalog (Version 1.0, February 2008).* Technical Report TB0801, Chair for Informatics 19 (sebis), Technische Universität München, 2008.

[CIO99] The Chief Information Officers Council: *Federal Enterprise Architecture Framework.* Version 1.1, 1999.

[Co04] Comella-Dorda, S.; Dean, J.; Morris, E.; Oberndorf, P.: *A Process for COTS Software Product Evaluation.* CMU Univ., SEI Institute, CMU/SEI-2003-TR-017, 2004.

[DD09] Department of Defense: *The Department of Defense Architecture Framework (DoDAF).* Version 2.0, 2009.

[De06] Dern, G.: *Management von IT-Architkturen - Leitlinien für die Ausrichtung, Planung und Gestaltung von Informationssystemen.* Vieweg, 2nd edition, 2006.

[Du94] Dumslaff, U.; Ebert, J.; Mertesacker, M.; Winter, A.: *Ein Vorgehensmodell zur Software-Evaluation.* In: HDM - Praxis der Wirtschaftsinformatik 175/1994, pages 89-105, 1994.

[FAW07] Fischer, R.; Aier, S.; Winter, R.: *A Federated Approach to Enterprise Architecture Model Maintenance.* In: EMISA 2007, pages 9-22, 2007.

[FJ02] Franck, E.; Jungwirth, C.: *Reconciling investors and donators - The governance structure of open source.* Working Paper 8, University of Zurich, Zurich, Switzerland, 2002.

[FP02] Ferrin, B.G.; Plank, R.E.: *Total Cost of Ownership Models: An Exploratory Study.* In: Journal of Supply Chain Management, Vol. 38(3), pages 18-29, 2002.

[Fr02] Frank, U.: *Multi-Perspective Enterprise Modeling (MEMO) - Conceptual Framework and Modeling Languages.* In: Proceedings of the 35th Annual Hawaii International Conference on System Sciences, pages 1258-1267, 2002.

[FW06] Fischer, R.; Winter, R.: *Essential Layers, Artifacts, and Dependencies of Enterprise Architecture.*. IEEE Computer Society: EDOC Workshop on Trends in Enterprise Architecture Research (TEAR 2006), Hong Kong, 16.10.2006, IEEE Computer Society, Los Alamitos, CA, USA, 2006.

[HW08] Hafner, M.; Winter, R.: *Processes for Enterprise Application Architecture Management.* In: Proceedings of the 41st Hawaii International Conference on System Sciences 2008, pages 396, 2008.

[IE00] IEEE: *Recommended Practice for Architectural Description of Software Intensive Systems (IEEE Std 1471-2000).* IEEE Computer Society, New York, 2000.

[Ke06] Keller, W.: *IT-Unternehmensarchitectur - Von der Geschäftsstrategie zur optimalen IT-Unterstützung.* dpunkt Verlag, Heidelberg, 2006.

[KH06] von Krogh, G.; von Hippel, E.: *The Promise of Research on Open Source Software.* In: Management Science, Vol. 52(7), pages 975-983, 2006.

[KM01] Kogut, B.; Metiu, A.: *Open source software development and distributed innovation.* Oxford Rev. Econom. Policy, Vol. 17(2), pages 248-264, 2001.

[KW07] Kurpjuweit, S.; Winter, R.: *Viewpoint-based meta model engineering.* In: Reichert, M.; Strecker, S.; Turowski, K.; eds.: Enterprise Modeling and Information Systems Architectures - Concepts and Applications. EMISA'07, pages 143-161, 2007.

[La05] Lankhorst, M.: *Enterprise Architecture at Work - Modeling, Communication and Analysis.* Springer, Heidelberg, 2005.

[Ma08] Matthes, F.; Buckl, S.; Leitel, J.; Schweda, C.M.: *Enterprise Architecture Management Tool Survey 2008.* TU München, Chair for Informatics 19, Prof. Matthes (sebis), 2008.

[MN98] Maiden, N. A.; Ncube, C.: *Acquiring COTS Software Selection Requirements.* In: IEEE Software, Vol. 15(2), pages 46-56, 1998.

[MRE07] Mohamed, A.; Ruhe, G.; Eberlein, A.: *COTS Selection: Past, Present, and Future.* In: Proceedings of the 14th Annual IEEE International Conference and Workshops on the Engineering of Cumputer-Based Systems (ECBS'07), 2007.

[MT97] Morisio, M.; Tsoukiàs, A.: *IusWare: a methodology for the evaluation and selection of software products.* In: IEE Software Engineering, Vol. 144 (3), pages 162-174, 1997.

[Ni05] Niemann, K. D.: *Von der Unternehmensarchitektur zur IT-Governance - Bausteine für ein wirksames IT-Management.* Vieweg Verlag, 2005.

[NM99] Ncube, C.; Maiden, N. A.: *PORE: Procurement-Oriented Requirements Engineering Method for the Component Based Systems Engineering Development Paradigm.* In: Intr. Workshop on Component Based Software Engineeing, Los Angeles, CA, 1999.

[Oc00] Ochs, M.; Pfahl, D.; Chrobok-Diening, G.; Nothhelfer-Kolb, B. : *A COTS Acquisition Process: Definition and Application Excperience.* In: ESCOM-SCOPE'00, Munich, Germany, 2000.

[OG09] The Open Group: *TOGAF Enterprise Edition.* Version 9, 2009.

[Ra99] Raymond, E.: *The Cathedral and the Bazaar.* O'Reilly, Sebastopol, CA, 1999.

[RWR06] Ross, J.W.; Weill, P.; Robertson, D.C.: *Enterprise Architecture as Strategy.* Havard Business School Press, Boston, Massachusetts, USA, 2006.

[Sc08] Schekkerman, J.: *Enterprise Architecture Good Practices Guide - How to Manage the Enterprise Architecture Practice.* Trafford Publishing, Victoria, Canada, 2008.

[se05] sebis: *Enterprise Architecture Management Tool Survey 2005.* TU München, Chair for Informatics 19, Prof. Matthes (sebis), Germany, 2005.

[Vl99] Vlahavas, I.; Stamelos, I.; Refanidis, I.; Tsoukiàs, A.: *ESSE: an expert system for software evaluation.* In: Knowledge-Based Systems, Vol. 12(4), August 1999, pages 183-197, 1999.

[We03] Wegmann, A.: *On the Systemic Enterprise Architecture Methodology (SEAM).* In: Proceddings of the 5th International Conference on Enterprise Information Systems, pages 483-490, 2003.

[Wi07] Wittenburg, A.: *Softwarekartographie - Modelle und Methoden zur systematischen Visualisierung von Anwendungslandschaften.* Dissertation, Fakutält für Informatik, Technische Universität München, 2007.

[Za87] Zachman, J.: *A framework for information systems architecture.* In: IBM Systems Journal Vol. 26(3), pages 276-292, 1987.

C.Appendix – List of criteria

1. Initiator data

1.1. Please provide the name of the company concerned with the development of the tool.

1.2. Where is the headquarter of the company located? Where does the company have subsidiaries?

1.3. When was the company founded?

1.4. How many employees does the company employ?

1.5. How many employees are concerned with developing the tool?

1.6. What was the motivation to release the tool as open source?

1.7. What are the main products of the company?

2. Community data

2.1. How many developers have joined the community?

2.2. How many of the community members can be described as active, i.e. contribute code regularly?

2.3. Please outline how decisions are made in the community. Is a core team or board guiding the decision making?

2.4. How often are code changes integrated?

2.5. Do you offer several versions of the tool? For instance, a stable version with fewer features and a nightly version with the latest features.

2.6. In which ways do you support community members (e.g. documentation, forum)?

3. Tool data

3.1. Please provide the name of the tool, including version numbers.

3.2. When was the current version released?

3.3. Please provide a brief history of the tool.

3.4. Please outline the schedule for the next minor and major releases of the tool and sketch the new functions in the upcoming version.

3.5. Under which license is the tool available? Is it an OSI approved license?

4. General tool architecture

4.1. Please provide an overview of the tool's infrastructure requirements (hardware, operating system, RDBMS, browser – if appropriate distinguish different aspects of the tool, e.g. thin client).

4.2. Which open source libraries/components are used by your tool?

4.3. What platforms or database systems (e.g. DB2, MySQL) does the tool support?

5. Collaboration support

5.1. Does the tool support multiuser work? Please provide information due to multiple reading and writing.

5.2. What kind of synchronization mechanism is provided to support multiple user edits, e.g. locking, timestamp based synchronization?

5.3. In case locking is supported, which locking modes (e.g. shared, exclusive) does the tool support and which locking granularities (e.g. whole models, diagrams, set of entities) are distinguished?

5.4. Does the tool provide rights management for restricting user's access, e.g. to models, diagrams or limit their editing capabilities concerning e.g. certain entities? Can users pass the rights given to them to others (with grant)?

5.5. Does the tool support versioning of artifacts in respect to collaboration support, i.e. can a model be reverted to the status prior to changes by a certain user?

5.6. Does the tool offer capabilities for offline working with the data, e.g. a client, from which edits can be synchronized with the repository? What kind of operations are supported on offline data?

5.7. Does the tool support multi-client capability to allow simultaneous access to several clients without seeing each other's data?

5.8. Does the tool support automatic notifications (especially when changing certain objects)?

5.9. Does the tool support substitution rights (e.g. vacation replacement)?

5.10. Does the tool support integration in corporate portals (e.g. wikis) to support collaboration? If yes, how is it implemented?

6. Internationalization/Localization

6.1. Does the tool provide capabilities to assign a locale to a user profile? Which adaptions to e.g. the graphical user interface of the tool may be defined in a locale (e.g. date format, currency)?

6.2. Does the tool support multi-language data, e.g. naming or description of entities dependent on the user's language within one installation/instance of the tool?

6.3. Does the tool support unicode characters?

7. Integration with related domains

7.1. Does the tool support business process modeling? Which business process modeling standards/notations does the tool support?

7.2. EPC, BPML, BSEL, WSCI, BPEL, other

7.3. Does the tool support data modeling? Which modeling standards/notations does the tool support?

7.4. E/R, Crowfoot notation, IDEF1X, UML with profiles

7.5. Does the tool support UML modeling? How many diagrams does the tool support? Which diagrams does the tool support?

7.6. class diagram, composite structure diagram, component diagram, deployment diagram, object diagram, package diagram, activity diagram, use case diagram, state machine diagram, sequence diagram, collaboration diagram, timing diagram

8. Meta model

8.1. Please provide information on the predefined metamodels shipped with the tool (number of classes, associations)?

8.2. Do the predefined metamodels comply with EA frameworks, as e.g. Zachman, TOGAF?

8.3. Please provide information on the number of classes (entity types) contained in the metamodels, especially of the standard or default metamodel employed for EAM.

9. Integration with other tools

9.1. Please provide information on different formats and tools, from which data can be imported into the tool, e.g. CVS, Excel, Microsoft Project. Please detail on how transformations for importing this data can be implemented or configured.

9.2. Does the tool support accessing data from a BPM tool via an interface? Which BPM tools are supported by which interfaces (offline, online)?

ARIS Toolset (IDS Scheer), ADONIS (BOC), Corporate Modeler (Casewise),…

9.3. Does the tool support accessing data from a CMDB via an interface? Which CMDBs are supported by which interfaces (offline, online)?

Atrium (BMC), Tivoli CMDB (IBM), CMDB (HP),…

9.4. Does the tool support accessing data from a systems management tool via an interface? Which CMDBs are supported by which interfaces (offline, online)?

Tivoli (IBM), OpenView (HP), SMS (Microsoft),…

9.5. Does the tool support accessing data from a project management tool via an interface? Which CMDBs are supported by which interfaces (offline, online)?

Clarity (CA), Mercury PPM (HP), BW (SAP), Project (Microsoft),…

9.6. Does the tool support accessing data from an ERP tool via an interface? Which CMDBs are supported by which interfaces (offline, online)?

SAP, Oracle,…

9.7. Please detail on the mechanisms for synchronizing and keeping consistency with data from an external data source?

9.8. Does the tool support a connection to workflow engines?

9.9. What kind of export formats will be supported? Is it possible to export XML?

9.10. Does the tool support single sign-on and can other existing user directories (like LDAP) be used? Does the tool support the OpenID standard?