A FRAMEWORK FOR PROJECT ARCHITECTURE IN THE CONTEXT OF ENTERPRISE ARCHITECTURE

Foorthuis, Ralph Marcel, Statistics Netherlands, Prinses Beatrixlaan 428, 2273 XZ Voorburg, the Netherlands, RFTS@cbs.nl

Brinkkemper, Sjaak, Utrecht University, Institute of Information and Computer Sciences, Padualaan 14, 3584 CH Utrecht, the Netherlands, S.Brinkkemper@cs.uu.nl

Abstract

Little scientific research has as yet been done on projects conforming to Enterprise Architecture. To lay foundations for such research, this paper presents a theoretical framework for defining the Project Architecture (PA) in the context of working with Enterprise Architecture. One part of the PA is the Project Start Architecture (PSA), which bounds the project to the Enterprise Architecture (EA) and/or Domain Architecture (DA). We start with explicating the context of a PSA in terms of its relation to the EA and DA. Subsequently, we define the PA in terms of three dimensions. The first dimension contains four aspect areas. The second dimension features four abstraction levels. The third dimension contains two project content categories: the PSA (containing prescriptions inherited from the EA and/or DA) and the PED (the Project Exclusive Design, containing the fundamental analysis and design artifacts that have been created specifically for the project). A real-life case is used to help illustrate and validate the theoretical framework. Additionally, a mapping with RUP artifacts is made to further clarify the framework of the PA with examples of well-known analysis and design artifact types.

Keywords: Enterprise Architecture, Project Architecture, Project Start Architecture, RUP, IAF

1 INTRODUCTION

The young field of Enterprise Architecture (EA) has attracted quite some attention over the last few years, and numerous articles have been published. Surprisingly, little scientific research has yet been done on the topic of carrying out projects conforming to an Enterprise Architecture. This, however, is a highly relevant research area, since EA is claimed to provide projects with value in a number of ways. Working with EA is said to improve project success, to reduce project risk, duration and complexity, to speed up the initialization of a project and to reduce project costs (Bucher et al. 2006, Wagter et al. 2005, Capgemini 2006). However, scientific evidence for these claims is still missing. Before research can validate them, clear definitions of architectural concepts at the project level are required. Miscommunication about terms as *Architecture*, *Project Architecture*, *Project Start Architecture* and *Software Architecture* is a risk both organizations and the scientific community run when discussing architecture in projects conforming to EA. Therefore, the research question in this paper is:

> *How can the various architectures at the project level, and their interdependencies and contents, be defined in the context of Enterprise Architecture?*

The goal of this research is to contribute to clear concepts of architecture when applying them within (research on) projects that should adhere to an EA. In our explorative study, we hope to start a discussion on the nature of Project Architecture (PA), Project Start Architecture (PSA) and other types of architecture at the project level. Furthermore, our aim is to lay foundations for – and to stimulate further research on – the topic of project conformance to Enterprise Architecture.

1.1 Overview

In section 2 we state our view on Enterprise Architecture and position the PSA in the context of this larger picture. In section 3 we present a three dimensional framework for the Project Architecture, of which the PSA will be one of the constituents. The framework will be illustrated by presenting examples from a real-life case and a mapping with well-known software engineering artifacts. In section 4 we explain the research approach we have applied for creating a real-life PSA, which we use to help illustrate and validate this part of our theoretical framework. In section 5 we state our conclusions.

2 VIEW ON ENTERPRISE ARCHITECTURE

The IEEE standard 1471-2000 defines *architecture* 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". We define *Enterprise Architecture* according to Bucher et al. (2006) as: (1) the fundamental organization of a government agency or a corporation, either as a whole, or together with partners, suppliers and/or customers, or in part (business units), as well as (2) the principles governing its design and evolution. Being on a high level, the EA focuses on the essentials of the enterprise, which, in contrast to specific solutions, are relatively stable over time. Also, the high-level view on the enterprise enables its management to pursue a strategy that is optimal for the company as a whole, instead of local optimizations. The EA, then, should provide an integrated and coherent view on the enterprise, aligning business, information and IT, and guiding specific projects. As a consequence, the EA assists in achieving the enterprise's essential business objectives (Lankhorst et al. 2005).

Prescriptions are used in EA to provide constraints and direction. As such, they are the means by which the EA (and possibly a Domain Architecture) influence projects. Prescriptions can take various forms. They can be text-based *principles* that state a generic requirement, or they can be graphical *models* that depict a generic process or structure which can be refined by the project which takes them as a starting point. Prescriptions will evolve but should be relatively stable over time.

2.1 Architecture framework

An architecture framework is a conceptual structure to analyse an enterprise and to structure both an architecture and its design process. Such a framework often takes the shape of a two-dimensional matrix. The cells in the matrix describe the content elements of the architecture and their relationships. The matrix therefore gives an overview and helps to identify required analysis or design artifacts, such as information models or documents containing principles. A number of architecture frameworks exist. As a starting point, we will use a framework based on the Integrated Architecture Framework, or IAF (Capgemini 2006, Macaulay 2004, Goedvolk et al. 1999). IAF is relevant for our research for several reasons. First, it features a well-known and widely accepted categorization of aspect areas. Similar or identical areas are used in academic research (Henderson & Venkatraman 1993, Pulkkinen 2006, Maes 2003) and by organizations offering EA services (The Open Group 2003, Sousa et al. 2004, Wagter et al. 2005). Second, for its vertical dimension it uses abstraction levels that can be applied very well on both enterprise and project level, making it possible to explicate the contents of the Project Architecture and to present examples of real-life principles and analysis and design artifact types. The IAF-framework is shown in Figure 1. Examples of content elements in the cells will be presented in subsequent sections.

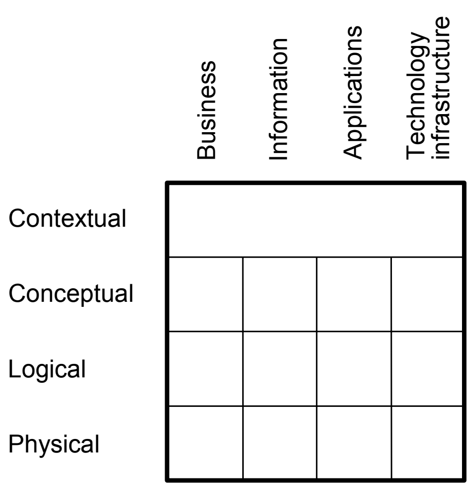


Figure 1. The IAF-framework for EA

On the horizontal dimension a categorisation of different *aspect areas* is used:

- *Business*: the relationship of the enterprise with its environment, the business objectives and strategy, the offered products and services, the organizational units and their relationships, the governance, the people, the key business processes and strategic projects.
- *Information*: the creation, processing, exchange, storage and use of information and knowledge. The structure of information elements and their relationships also belongs to this area.
- *Applications*: the (network of) IT-systems which offer communication and information services for the business and information areas. This includes both off-the-shelf and tailor-made systems.
- *Technology infrastructure*: the (network of) hardware devices, operating systems and middleware on which the applications run and which deliver processing, transmission and storage capabilities.

On the vertical dimension abstraction levels detail issues identified on higher levels:

- *Contextual*: this level focuses on the context in which the organization resides, the vision and the business goals. Also, it explains why the Enterprise Architecture is created and states its scope and highest-level principles. The Contextual level is considered to stand above the aspect areas.
- *Conceptual*: the identification of all EA requirements, without detailing how they will be realized.
- *Logical*: this level focuses on designing an ideal solution. It details how the requirements can be realized, but in an implementation independent way.
- *Physical*: this level focuses on translating the logical ideal EA solution into an implementation specific real world version. This level, therefore, provides standards, guidelines and generic specifications for the detailed design or selection of solutions to be developed or purchased.

2.2 Kinds of architectures in working with EA

When working with Enterprise Architecture (EA), one can distinguish between different kinds of architectures. First, there is the *EA* itself, which is the architecture residing on the level of the enterprise. Second, if needed, one or more *Domain Architectures* (DAs) can be created. These are architectures defined on the basis of one specific group of products, services, processes or functions. A domain can be acknowledged on the level of the enterprise, for example security or a specific process step that is used throughout the organization. However, a DA can also reside below enterprise-level, for instance when creating guidelines for one specific product group. As a third kind, *Project Start Architectures* (PSAs) can be distinguished. An architecture on the level of the enterprise or a domain does not detail the complete functional and technical design for a specific solution. This will be done in a lower level project that should conform to the general EA and/or relevant DA. Therefore, at the start of such a project, a PSA can be created (Wagter et al. 2005). This is an architecture on the project level that inherits and, if needed, translates the prescriptions from the EA, and possibly a DA, to prescriptions that are tailored to the specific project at hand. The project, subsequently, must further detail and design the proposed solution within the specified boundaries of the PSA. An important part of the PSA is taking into account the links the project has with outside elements, like other projects, organizational units and enterprise-wide standards. A specific project will have only one PSA. In Figure 2, the arrows show that prescriptions of an architecture affect other architectures. Note that we consider feedback loops to be important, but beyond the scope of this paper. In the next section, we will zoom in on the PSA and the other constituents of the Project Architecture.

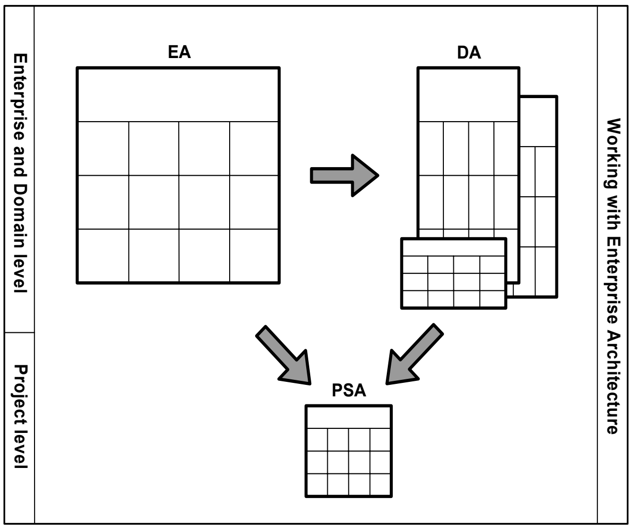

EA
DA
PSA
Enterprise and Domain level
Project level
Working with Enterprise Architecture

Figure 2. Architectures and flow of prescriptions

3 THE FRAMEWORK FOR PROJECT ARCHITECTURE

The PSA provides the constraints and general direction for the further elaboration of the project's fundamental design. The combination of the PSA and the remaining core design of the project constitutes the Project Architecture (PA). See Figure 3 for a graphical representation. The PA consists of the essential design elements of a specific project solution and their guiding principles. In other words, a PA is the design of the fundamental elements of an IT-system and the core business processes that it will support, adhering to the relevant EA and/or DA prescriptions. The practical use of acknowledging the Project Architecture is that the analysis and design artifacts it contains can be sent to the enterprise or domain architects. This way, the actual project design can be checked on consistency with the EA and/or DA during and after the project. We will elaborate on the PA by adding to the framework of section 2 a third dimension, which contains project content categories. We distinguish between two categories:

- **Project Start Architecture (PSA)**: The collection of prescriptions from an EA and/or DA that is relevant for the specific project.
- **Project Exclusive Design (PED)**: This category contains the fundamental analysis and design artifacts that are created for the specific project.

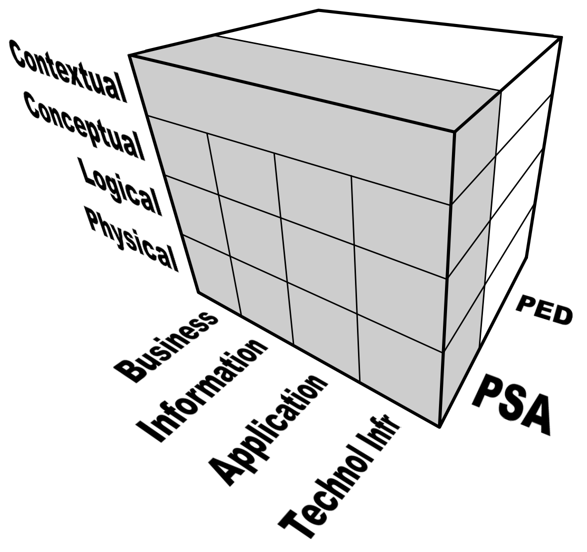


Figure 3. The framework for Project Architecture

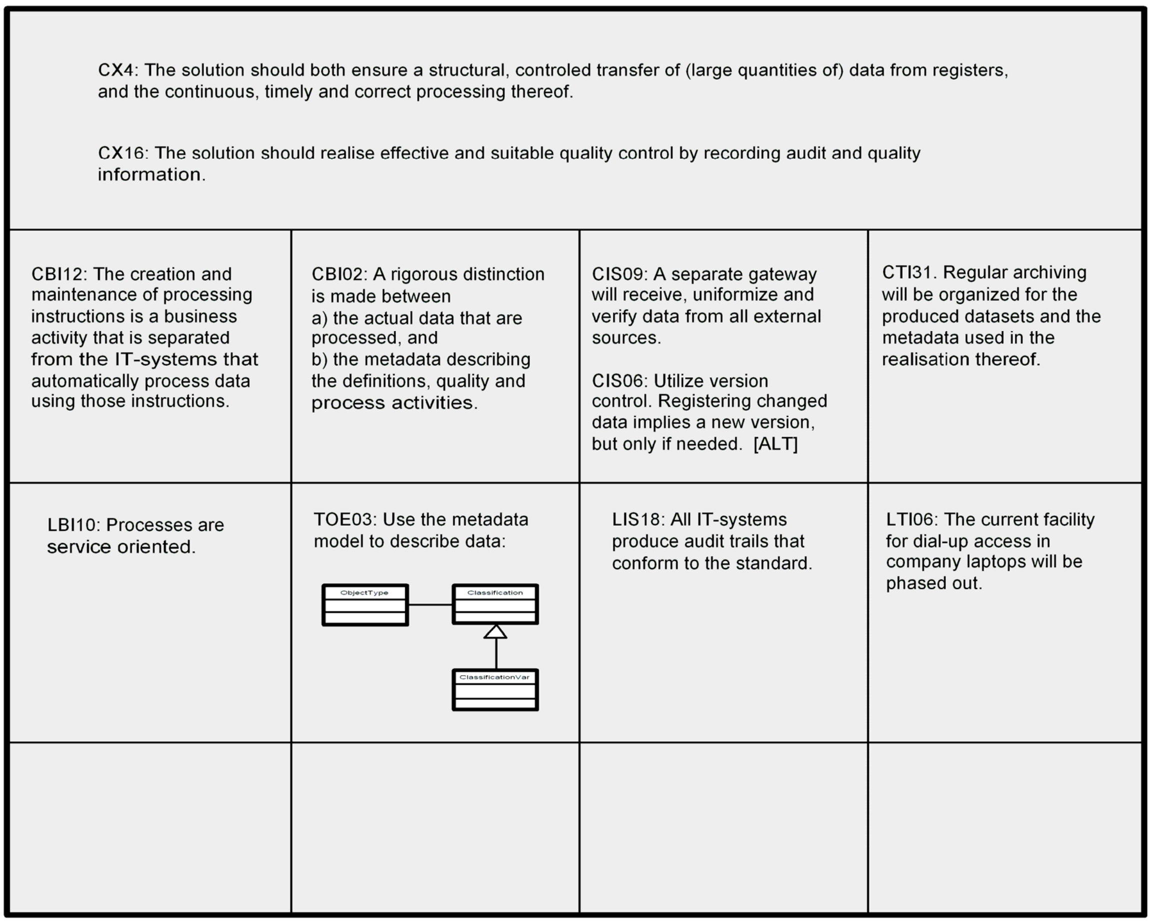
The figure above shows that the PA could also be defined as the combination of the PSA and the PED. There are of course important connections between these two content categories. First, one can distinguish between temporal phases in the project: the prescriptions of the PSA should be identified at the start of the project, after which the PED can be created. Second, the PED should conform to, and refine the contents of the PSA. Note that even a PA does not provide a complete view on the entire project. The PA focuses on the fundamentals, or core elements, of the project. The cube therefore contains architecture-related artifacts.

3.1 Dicing the grey slice: the PSA

The greyed category on the project content dimension represents the PSA. In this section we will elaborate further on this and present examples from a real-life PSA, which inherited text-based principles and graphical models from its EA and DA. Our definition of the Project Start Architecture is based on descriptions given by Wagter et al. (2005). A PSA is an architecture at project-level that inherits from the EA and/or DA those prescriptions that are relevant to the project and, if alteration is needed, translates them to specific project prescriptions. Also, additional prescriptions can be added. A PSA should be created at the start of the project, a process in which enterprise and/or domain architects can assist. The PSA is the mechanism that links the project to the EA and/or DA. The value of a PSA should lie in ensuring that the project conforms to the EA and DA, for it sets the boundaries within which the project should operate. As such, the PSA can be seen as an agreement. In addition, it should allow for a quick start, since at the beginning of the project several fundamental decisions have already been taken and alignment with other projects has been taken care of (Wagter et al. 2005). The PSA will contain a relatively small body of content. This should make it easy to read for (new) project members, although some jargon will probably be included.

The PSA can be seen a collection of prescriptions. It is useful to clearly distinguish between prescriptions that have been taken from the EA and/or DA literally and prescriptions that have in some way been altered or added. The reason for this is that the enterprise or domain architects that have to review and maybe even approve the PSA, will definitely want to take a look at the non-original prescriptions. In contrast, the prescriptions that have been taken from the EA and/or DA literally will most likely not be highly controversial. In order to make it easier for the reviewers to find specific types of prescriptions, we have developed a set of *labels*. Every prescription can have one of the labels explicitly attached to it, stating its type.

- **APL**: Directly Applicable Prescription. This prescription is inherited literally from the EA or DA and no major problems are expected in applying it.
- **ALT**: Altered Prescription. This is a prescription in the PSA coming from the EA or DA, but has to some degree been modified in order to fit in with the project.
- **ADD**: Added Prescription. This prescription is created specifically for the project at the very start of it. Note that this renders the PSA quite extensible. Such a prescription might also be a candidate for being included in the EA or DA.
- **AMB**: Ambiguous Prescription. This is a prescription that is still subject to debate concerning its implications for or relevancy to the project. This kind of prescription should not be present in the final version of the PSA. For a quick look-up during review sessions, however, it can be helpful to explicitly acknowledge them in earlier versions of the document.
- **ABD**: Abandoned Prescription. This is a prescription that at one point might have seemed relevant to the project, but has been dropped eventually. For documentation purposes such a prescription, and the reasons for dropping it, can be explicitly included in the PSA.



CX4: The solution should both ensure a structural, controled transfer of (large quantities of) data from registers, and the continuous, timely and correct processing thereof.

CX16: The solution should realise effective and suitable quality control by recording audit and quality information.

CBI12: The creation and maintenance of processing instructions is a business activity that is separated from the IT-systems that automatically process data using those instructions.	CBI02: A rigorous distinction is made between a) the actual data that are processed, and b) the metadata describing the definitions, quality and process activities.	CIS09: A separate gateway will receive, uniformize and verify data from all external sources. CIS06: Utilize version control. Registering changed data implies a new version, but only if needed. [ALT]	CTI31. Regular archiving will be organized for the produced datasets and the metadata used in the realisation thereof.
LBI10: Processes are service oriented.	TOE03: Use the metadata model to describe data:	LIS18: All IT-systems produce audit trails that conform to the standard.	LTI06: The current facility for dial-up access in company laptops will be phased out.

Figure 4. The PSA

The matrix above features the aspect area and abstraction level dimensions for the PSA project content category. We have included prescriptions from a real-life case to illustrate the various cells. Prescriptions marked with [ALT] have in some way been altered from its original formulation in the EA or DA. See section 4 for more information about the empirical case.

3.2 Dicing the white slice: the PED

In order to further clarify what can be regarded as being part of a PA, but not of a PSA, we have cut out the relevant slice in order to discuss it in more detail. The PED contains on the fundamental analysis and design artifacts of a project that should conform to and refine the PSA, but are not included in it. Since the elements that make up this frame are entire documents and models, it is not possible to give real-life examples of elements in the cells. However, we can give examples of analysis and design artifact types of the Rational Unified Process (RUP). This is a software engineering process that provides a disciplined approach to assigning tasks and responsibilities when developing software (Kruchten 2003, Rational 2003). The mapping of RUP artifacts on the framework should make more tangible what constitutes the PED.

One part of the PED is the project's *Software Architecture* (SA), which is "the fundamental organization of an information system embodied in its components, their relationships to each other, and to the environment, and the principles guiding its design and evolution". As can be seen from the Software Architecture Document (SAD) artifact in Figure 5, the SA is located in the four lower right cells of the Logical and Physical layers of the Application and Technology Infrastructure areas. The Software Architecture itself is a specific solution design that is not considered to be part of working with an EA, DA or PSA (although prescriptions might provide constraints and direction for its design).

Business Vision document Vision document			
Business Use Case Model Business Use Case Specifications (that describe key processes)	Information areas List of domain entities	Use Case Model Use Case Specifications (that are relevant for the software architecture) Supplementary Specification	Supplementary Specification
Business Use Case Realizations (that describe key processes) Statistical methodology	Domain Model	Software Architecture Document (e.g. Logical View) Logical database model Use Case Realizations (that are relevant for the software architecture)	Software Architecture Document (e.g. Deployment View)
		Software Architecture Document (e.g. Implementation View) Physical database model	Software Architecture Document (e.g. Deployment View)

Figure 5. The PED

The cells in the matrix contain examples of RUP analysis and design artifacts that are created specifically for a project. One example is the Conceptual-Application cell, which deals with *what* the information system should offer its stakeholders. This concerns the functional requirements, which in RUP are described in detail in Use Case Specifications. Another example is the Logical-Information cell, which not describes *what* information elements the business has (this is described in the list of domain entities in the upper cell), but *how* these entities are structured and relate to each other. A final example is the Physical-Application cell, which deals with the technical fundamentals of the information system. Examples of artifacts that make up this cell are the Implementation View of the SAD and the physical data model.

The artifacts in the PED can contain a substantial amount of content, depending on the size and complexity of the project. The PSA slice will undoubtedly be smaller. Therefore, when offering the PED to enterprise or domain architects for review or approval purposes, it would be practical to work with *views*. For instance, the project can decide to offer only the artifacts in the Application column to a technical reviewer, or only the artifacts in the Contextual layer to an information manager.

3.3 Some remarks about the Project Architecture

Since the PA focuses on the fundamentals of the project, the cube contains *architecture-related* content. Therefore, several artifacts that will be created in a project are still missing. For example, Use Cases that are not relevant for the SA are excluded. Also, detailed technical design of components or user interfaces is not considered to be part of the PA. The same holds true for the programming code. Furthermore, the PA does not include the end-products, e.g. the executables, databases and manuals. An interesting aspect of the Project Architecture framework is that it brings together two different types of architecture. First, the high-level *Enterprise* and *Domain Architecture* (in the PSA) and second the *Software Architecture* (in the PED). The SA is a totally different type of architecture, not covering all the cells of the IAF-framework and being valid only in the specific project.

4 EMPIRICAL CASE

A PSA for a real-life project was created by the principal researcher to test whether the framework could be applied in practice. An Action Research (AR) approach was used to carry out this field research. Since AR allows for studying technology in its human context (Baskerville et al. 1996), it was ideally suited to verify whether the PSA-part of our theoretical framework could be applied in a practical situation. To ensure relevance and scientific rigor, we consistently applied the principles of Canonical Action Research, as stated in Davison et al. (2004). Data was gathered by recording and/or taking minutes of discussions, keeping a daily research diary and analysing documents (e.g. EA artifacts).

4.1 Research setting

We carried out our research in Statistics Netherlands (CBS), a large governmental organization located in two cities in the Netherlands, employing about 2000 people. Its mission is to produce and publish undisputed, coherent and relevant statistical information. The organization is information intensive in nature, since both its input and output are information. Six months prior to the start of our research, the EA of the organization had been officially approved by high-level management, which meant that working with EA was relatively new to the organization. Although the organization planned to utilize the concept of the PSA, no formal methods or templates had yet been defined for this purpose.

The EA itself consists of five central documents (258 pages), containing the prescriptions, plus some supportive material. The EA focuses on providing a complete architecture, although, at the time of research, the physical layer did not contain any prescriptions yet. In total, a number of 247 text-based principles had then been formulated, which were uniquely identified by an ID code (e.g. "CX16"). In addition to the principles, 75 graphical models were included, some of which were high-level designs (e.g. of business processes), while others had a more clarifying purpose (e.g. showing the context of the organization). In addition, the EA documents contained descriptive text, for commenting on and relating the principles and models. In presenting the prescriptions, the Business and Information areas of the EA were joined into one column, the reason for which was the fact that the main input and output of this organization consisted of information. As a consequence, in creating the EA it had proven difficult to reach a widely accepted distinction between business products and information objects.

4.2 Results

The researcher participated as a business analyst in the first phase of a project that had an expected duration of several years. The goal of the project was to redesign the business processes and IT-systems of one of the organization's key statistical products. In the business analysis, a PSA was created, in which the researcher had considerable freedom since no formal approach in working with a PSA had been introduced yet. Most of the prescriptions originated from the EA. However, at the time of drawing up the PSA, a Domain Architecture for the central storage and retrieval of metadata was being created. Although this DA was not yet finished and no principles could be inherited from it, its central metadata model was considered useful in the project since it provided a means to describe the business objects. Also, it was thought that using this model should allow for relatively easy alignment between the current project and the completed DA at a later stage.

Three weeks and four iteration cycles were needed to create the final PSA, the format of which was based on the theoretical framework presented in this paper. The primary researcher participated in all iterations. A standard iteration would consist of analysing the shortcomings of the current version of the PSA, creating a new version, distributing it to the relevant people, and organizing a review session. The final PSA, counting 21 pages, was accepted by the project members, the involved enterprise and domain architects and the project board. The four iterations were needed to clarify the text-based principles by adding comments, to add a requested section about the feasibility of adhering to the prescriptions, to add extra principles and to add the metadata model from the DA. In the process, discussions were held with project members, enterprise and domain architects and another business analyst.

The PSA featured a heavy focus on the Business and Information area since the project started out as a business redesign project. It was decided that the Application and Technology Infrastructure areas of the PSA would be supplemented with prescriptions from the corresponding areas of the EA when sub-projects would start to build actual IT-solutions (which could take a year). The table below features an overview of the number of text-based principles (p) and graphical models (m) that were actually used in the PSA. The number of prescriptions in the EA is specified between square brackets.

Contextual	p: 12 [31] m: 0 [4]		
Conceptual	p: 15 + 2 ADD [19] m: 1 from EA [10]	p: 7 + 1 ALT [37] m: 0 [3]	p: 8 [52] m: 0 [10]
Logical	p: 10 + 1 ADD + 1 ALT [32] m: 1 from DA [9]	p: 3 + 1 ALT [25] m: 0 [10]	p: 3 [51] m: 0 [21]
Physical	-	-	-
	Business & Information	*Application*	*Technol Infr*

Table 1. Prescriptions in the PSA [and EA]

In the PSA that was created, the principles were presented in a table, with one column containing their unique ID's. A second column was used for the principle statement and its comments. A third column contained the label that the principle had been given (e.g. ADD or ALT). If the principle was inherited literally from an EA or DA, this column remained empty in order to focus the attention of the reviewer to the ambiguous, altered, added or abandoned principles. This was considered practical by the people involved, for in a review process these latter types are expected to be the most controversial, and therefore the most important ones to be discussed. Every cell in the PSA ended up containing several text-based principles (except for the cells on the physical level, as a result of the lack of prescriptions here). Since the Business and Information areas of the EA were combined, they were also combined in the PSA of the project. However, for illustration purposes we classified the examples in Figure 4 as belonging to one of these areas. At the time of writing this article, the PED was not yet completed.

Since only the relevant prescriptions were added, in a relatively short document, the project members were able to read the PSA, which stimulated discussion. Discussing this PSA several times with the project members proved to have some advantages. First, it created EA-awareness in an early stage of the project. Second, discussing the prescriptions created richer and more tangible insight in the EA, and the consequences that it would have on the project.

Interestingly, a significant level of interpretation was possible, and indeed required, when discussing the meaning and implications of applying prescriptions. This was due to the fact that the EA provided abstract, generic prescriptions at enterprise level. For example, principle LBI10 in the Business and Information area stated that "processes are service oriented". However, the EA itself did not specify what defines a service oriented business process. In addition, the EA was partially created by external consultants, whom had left the organization at the time of creating the PSA. As a result of the interpretative nature, we used the prescriptions' comments section extensively for explicating our translations.

The learning during the AR-research, both for the organization and the researchers, took place mainly during the sessions that were organized to discuss the current version of the PSA. We have learned that the theoretical framework can be practically applied to a specific project when drawing up a PSA. The participants considered the abstraction levels, aspect areas and labels for categorizing prescriptions particularly practical instruments. The original theory did not feature the label for ambiguous (AMB) prescriptions. They were added to the framework as a result of the research, for it proved practical to be able to quickly find them during discussions and review sessions. The abandoned (ABD) prescriptions were also added to the theory as a result of discussions, although this type was not required for this particular PSA. The experiences with working with a PSA in the project were sent to the lead architects, so that they would be able to use the insights for an enterprise-wide PSA template.

5 CONCLUSIONS AND FURTHER RESEARCH

We have presented a theoretical framework that describes different architectures at project level: Project Architecture, Project Start Architecture and Software Architecture. The PA and PSA are architectures that have to be understood in the context of EA and can be described using the IAF framework. Software Architecture is a different type of architecture, one that can exist without an EA. It can, however, be influenced by an EA (and/or DA). For the PSA, a set of labels is presented which can be used to explicitly state the type of a prescription. A real-life case has been presented to illustrate the framework and show that it can be used to create a PSA, based on the prescriptions from an EA and/or DA. RUP artifacts have been mapped on the framework to illustrate the PED.

Partly because of the high abstraction level of the inherited prescriptions, quite some interpretation and discussion with architects and project members turned out to be required in order to understand and link them to the project. For this reason, the comments section of the prescriptions was used extensively to make the prescriptions and their implications for the project more understandable and tangible. A useful effect of the discussions was that they created architectural awareness in the project team.

Further research is needed to verify the framework in other projects and to study whether other EA-frameworks than IAF can be used to structure and describe the various architectures at project level. The reason for this is that it might very well be necessary to be able to choose between several frameworks, depending on the project specific situation. For example, in this paper we implicitly presume there is no difference between the public and the non-public sector, which might not necessarily hold true after empirical study. EA-adherence of projects using a PSA can also be studied. More in general, we hold the opinion that more research is needed on the interesting and relevant topic of utilizing Enterprise Architecture at project-level. We hope our study will stimulate further research in this area.

Acknowledgements The authors wish to thank Rik Bos, Frank Hofman, Marlies van Steenbergen, Wiel Bruls, Gerrit Zijlmans and the TEAR2007 reviewers for their helpful comments on the paper.

References

Baskerville, R.L., Wood-Harper, A.T. (1996). A critical perspective on action research as a method for information systems research. Journal of Information Technology, 11, 235-246.

Bucher, T., Fisher, R., Kurpjuweit, S., Winter, R. (2006). Enterprise Architecture Analysis and Application. An Exploratory Study. EDOC Workshop TEAR. URL: http://tear2006.telin.nl

Capgemini (2006). Architecture and the integrated architecture framework. URL: http://www. capgemini.com/services/soa/ent_architecture/?d=4C418BF7-2926-B6FF-8A63-7FAED70CCF8B

Davison, R.M., Martinsons, M.G., Kock, N. (2004). Principles of canonical action research. Information Systems Journal, 14, 65-86.

Goedvolk, J.G., de Bruin, H., Rijsenbrij, D.B.B. (1999). Integrated Architectural Design of Business and Information Systems. The Second Nordic Workshop on Software Architecture (NOSA'99).

Henderson, J.C. & Venkatraman, N. (1993). Strategic alignment: Leveraging information technology for transforming organizations (Reprint). IBM Systems Journal, 1999, Vol. 38, No 2/3, 472-484.

IEEE Computer Society. (2000). IEEE Recommended Practice for Architectural Description of Software-Intensive Systems. IEEE Standard 1471-2000, Sept. 21, 2000.

Kruchten, P. (2003). The Rational Unified Process. An Introduction. Third Edition. Boston: Addison-Wesley.

Lankhorst, M. et al. (2005). Enterprise architecture at work. Modelling, communication and analysis. Berlin: Springer.

Maes, R. (2003). On the alliance of executive education and research in information management at the University of Amsterdam. Intern. journal of information management, Vol. 26, Issue 3, 249-257.

Macaulay, A. (2004). Enterprise Architecture Design and the Integrated Architecture Framework. Microsoft Architects Journal, January 2004, No 1, 4-9.

The Open Group. (2003). TOGAF. Version 8.1 "Enterprise Edition". URL: http://www.opengroup.org/architecture/togaf8-doc/arch/

Pulkkinen, M. (2006). Systemic Management of Architectural Decisions in Enterprise Architecture Planning. Four Dimensions and Three Abstraction Levels. Proceedings of the 39th Annual Hawaii International Conference on System Sciences (HICSS'06), Track 8, p. 179a.

Rational. (2003). Rational Unified Process. Version 2003.06.00.65. Rational Software Corporation.

Sousa, P., Marques Pereira, C., Alves Marques, J. (2004). Enterprise Architecture Alignment Heuristics. Microsoft Architects Journal, October 2004, No 4, 34-39.

Wagter, R., Berg, M. van den, Luijpers, J., Steenbergen, M. van. (2005). Dynamic Enterprise Architecture: How to Make It Work. Hoboken, New Jersey: John Wiley & Sons.