Communication of IT-Architecture

De communicatie van IT-Architectuur
(met een samenvatting in het Nederlands)

Proefschrift

ter verkrijging van de graad van doctor aan de
Universiteit Utrecht op gezag van de rector magnificus,
prof. dr. J.C.Stoof, ingevolge het besluit van het college
voor promoties in het openbaar te verdedigen op
woensdag 24 september 2008 des middags te 2.30 uur

door

Hendrik Koning

geboren op 24 maart 1953 te Echt.

Promotoren: Prof.dr. S. Brinkkemper
Prof.dr. J.C. van Vliet

Co-promotor: Dr. R. Bos

This thesis was (partly) accomplished with financial support from Belastingdienst, Ordina Public Panfox, ING and Ordina Institute. We acknowledge DANS for their financial contribution towards the printing of this thesis.

SIKS Dissertation Series No. 2008-22

The research reported in this thesis has been carried out under the auspices of SIKS, the Dutch Research School for Information and Knowledge Systems.

Cover design: Lidwien van de Wijngaert

ISBN/EAN: 978-90-5335-163-5

Content

Acknowledgements

Though working on my thesis was for the most part a lonely adventure, there are a number of people I wish to thank for their indispensable contribution along the way.

I am grateful to Hans van Vliet for being my supervisor for almost four years. He gave me the opportunity to make a start in the academic world and he introduced me to the art of writing scientific papers, an art I am still trying to improve. I thank Hans de Bruin for answering all my newby questions. I thank Claire Dormann for showing a genuine interest in creating better IT-architecture diagrams and by suggesting books and references. With Johan Hoorn I had good discussions about research methods. Elly Lammers was always helpful to give practical assistance. Danny Greefhorst gave the subject of architecture frameworks an extra dimension by his enthusiasm and openness. I am grateful to Sjaak Brinkkemper and Rik Bos for being my supervisors for the past years. Thank you for taking my research efforts seriously and being available for me for discussion or needed comments on my papers. In the person of Wilke Schram I would like to thank the Department of Information and Computing Science at the Utrecht University for their hospitality.

I want to thank the companies that supported my research. At the Belastingdienst I had a good contact with Jack Krijnen. From Ordina Panfox Public a trio took turns: Jan Campschroer, Herm Wijgergangs and Hans Rob de Reus. With Jan I had interesting exploratory discussions about 'our' profession. A special thanks to Walter Bakker and Paul Laterveer of ING for all the years we have worked together!

I started this research in part time while working for ID Research (later Ordina Institute). A word of thanks to Gerard Wijers, Patrick Strengers, Hans Bosma and Dennis Verhoef. Then I switched jobs to SERC of which I want to name Gert Florijn for his enthusiasm on architecture in software engineering.

My wife Else deserves a special place in this list. She had unshakeable confidence in the outcome of my research. She showed real interest and many times contributed by her remarks. She has been my soul mate for many years.

I thank Jesus Christ, the Son of God, my Lord and Saviour, who hears my prayers and sustains me day by day.

Curriculum Vitae

Henk Koning was born in Echt on March 24 of the year 1953. He did his secondary school in Eindhoven at the Eindhovens Protestants Lyceum, after which he studied mathematics at the Vrije Universiteit from 1970 until 1977 (cum laude). For three years he was a mathematics teacher at a secondary school, and then he switched to industry. He worked for the software house Volmac for four years, first as teacher of structured programming, later as system developer for various clients. For 16 years he worked with AEGON Insurance, first as researcher/architect at a central edp department in support of system developers, later as system developer / architect for the business unit AEGON Schade Bedrijven. The next two years he was an IT-consultant with Ordina Institute, and another two years in the same role with SERC (Software Engineering Research Center). At the moment Henk is employed by the Royal Netherlands Academy of Arts and Sciences (KNAW) as a system developer and project leader.

List of publications

- Koning, H., Dormann, C., van Vliet, H., 2002. Practical Guidelines for the Readability of IT-architecture Diagrams, in: Proceedings ACM SIGDOC 2002, pp 90-99.

- Koning, H., van Vliet, H., 2006a. Real-life IT architecture design reports and their relation to IEEE Std 1471 stakeholders and concerns, Automated Software Engineering (ASE) Journal, Volume 13, Issue 2, pp. 201-223.

- Koning, H., van Vliet, H., 2006b. A Method for Defining IEEE Std 1471 Viewpoints, the Journal of Systems and Software, volume 79 no 1, pp. 120-131.

- Koning, H, Bos, R, Brinkkemper, S, 2006. An Inquiry Tool for Stakeholder Concerns of Architectural Viewpoints: a Case Study at a large Financial Service Provider, TEAR2006 workshop of the 10th IEEE International Enterprise Distributed Object Computing Conference (EDOC 2006) ,16-20 October 2006, Hong Kong, China.

- Greefhorst, D, Koning, H, Van Vliet, H, 2006. The many faces of architectural descriptions, the journal of Information Systems Frontiers, volume 8 number 2, pp. 103-113.

- Koning, H, Bos, R, Brinkkemper, S, 2008. A Lightweight Method for the Modeling of Enterprise Architectures: Introduction and Usage Feedback, submitted for publication.

Abstract

This PhD thesis contains the results of various research activities that fall under the topic ‘communication of IT-architecture’. The term IT-architecture defines the various types of architecture that can be found in the domain of Information Technology (software architecture, enterprise architecture, etc).

Our overall conclusion is that good communication of IT-architecture is a matter of “meaningful structuring”. This has been worked out in de following sub topics.

A collection of 158 guidelines is presented to improve the readability of IT-architecture diagrams. These guidelines cover visual attributes like layout, hierarchy, colour, lines, graphics and text. Additional guidelines address the design of diagrams and give support for integrating diagrams in text.

IEEE Std 1471 proposes a conceptual model for documenting IT-architecture. Central concepts are ‘stakeholder concern’ and ‘views’. For four real life, pre-IEEE 1471, IT-architecture documents we investigated the pattern of relevancy to the stakeholder concerns. For each document a table was compiled that shows the relation between the parts of the document and the concerns of the stakeholders, as perceived by the authors. These tables show scattered patterns. For a stakeholder concern often only 25% - 50% of the document is relevant. The patterns show no evident way to convert the documents into IEEE 1471 views. We conclude that a structure of IEEE 1471 views needs to be incorporated in the setup of an IT-architecture description right from the start.

To achieve this, we propose a method to define IEEE 1471 viewpoints. The method consists of four steps: compile stakeholder profiles, summarize architectural design, relate the summary to the concerns of the stakeholders, and define viewpoints. For each step support is offered in the form of Word templates or Visio diagrams.

The IT-architects of one of the companies that took part in our research indicated that they did not like to define their own viewpoints, but rather work from available library viewpoints. To produce these for them, a round of stakeholder interviews was designed and an inquiry tool was compiled to solicit concerns (topics that are meaningful to the stakeholders). The tool is a questionnaire that covers a range of strategic IT-aspects. The tool was used to evaluate the existing architectural documentation practice.

So many architecture frameworks have been proposed in the past 15 years. We wondered which lessons could be learned from them. Based on an overview of 23 architecture frameworks we present nine base dimensions that structure collections of architecture documents: Type of information, Scope, Detail level, Stakeholder, Transformation, Quality attribute, Meta level, Nature and Representation. Architectural information is most often structured in two dimensions: one dimension addresses the type of information, and a second one has a sequential order.

Finally, for easy communication a lightweight Enterprise Architecture Modeling method is presented, based on these key architectural concepts: Enterprise, Information flow, Enterprise function, Flow of products & service, Scenario step, Application, Computer, Network. EAM structures the information in five diagram types: Supply Chain Diagram, Enterprise Function Diagram, Scenario Overlay, Application Overlay and System Infrastructure Diagram.

1 Introduction

Welcome to the thesis on "Communication of IT-architecture". This research is about IT-architecture documents. These documents contain high-level designs of new, or to be renewed, parts of an information technology (IT) configuration (hardware and/or software). IT-architectures are very important steering documents in IT-projects. It is crucial that the vision contained in IT-architecture documents is communicated well to all stakeholders.

This chapter introduces the concepts of 'IT-architecture' and 'communication' and poses the question 'what is communication of IT-architecture?'. After that the research questions and the research setting are introduced, and we discuss the research methods used and IEEE Std 1471. The chapter finishes with introductions to the following 6 chapters which contain the publications of this research.[1]

1.1 What is the problem?

Prior to formally working out our problem statement, we would like to sketch in an informal way what problems this research tackles. We do this in order to enable the readers of this thesis to decide whether they can identify with the problems raised and (partly) solved. Often heard complaints about architecture descriptions are: the information is too technical, the information is too much for what I need, it is not accessible or not up-to-date. Parnas and Clements (1986) mention these problems about software design documentation: poor organization, boring prose, confusing and inconsistent terminology and documenting the details instead of the major design decisions. This research focuses on the organization aspect, which is for architecture descriptions nowadays a matter of choosing the right 'views'.

More recently Clements et al (2003), pages 24 - 29, propose these guidelines for sound documentation of software architecture: write from the reader's point of view, avoid unnecessary repetition, avoid ambiguity, use a standard organization, record the rationale, keep it current but not too current and review it for fitness of purpose. We add to this that where these guidelines are not adhered to the documentation loses its usefulness and system development and maintenance problems can be expected. Bass et al. (2003, p 201) even state that without proper documentation an architecture becomes unusable. In this thesis you can find an inquiry tool to discover the 'reader's point of view'.

Clements et al. offer a number of reusable architectural views to aid practitioners in the field, but in practice the need is sometimes felt to create specific, situation dependent views. Rüping (2003, p 29) states in this respect "The most effective approach towards documentation is for each project to define its documentation requirements individually". This research offers support to enable the software specialist to create, where necessary, the proper views himself.

In a broader scope, over the years a lot of frameworks have been proposed to improve the quality and manageability of collections of architecture descriptions, but they differ much in terminology and setup. Instead of giving support they can augment the confusion. One author even wrote a book with the title "How to survive in the jungle of Enterprise Architecture Frameworks" (Schekkerman, 2004). We have compared a number of existing frameworks and discovered some insightful trends.

Another way of strengthening the work of describing architecture is by proposing conceptual modeling languages, sometimes accompanied by software tools. The problem one encounters here is that currently these languages are complicated and need a long learning

[1] In case you are reading this in a black and white print: some of the chapters (2, 7 and 9) contain color figures; on the homepage of this research (http://www.cs.uu.nl/people/koningh) and in the Igitur Archive (http://www.igitur.uu.nl/en/default.htm) the electronic version can be found with all the original colors.

curve. A new orientation on a simplified approach is necessary, which we have worked out in EAM (enterprise architecture modeling method).

This research has everything to do with aligning architectural information to the language (text and image) and needs (interests) of the readers of the information.

We will return to these topics, with proper references, in this and the following chapters.

1.2 What is IT-architecture?

Architecture is a relatively new branch within IT. The recent IEEE Std 1471 (IEEE, 2000) defines it as "Architecture is 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". Architecture is also described as ‘the earliest design decisions’, or ‘those aspects that are the hardest to change’ (Klusener et al, 2005). The Software Engineering Institute (2008) maintains a webpage with many definitions of (software) architecture. Architecture descriptions are high level design documents that guide the investment of often millions of dollars. Common components of architectural descriptions are: levels or domains which divide the area of operation into manageable parts, various types of components (user interface components, security components, data management components, etc.), complex relations between and behaviour of all these components to deliver together the required functionality, ways to build/buy/maintain the components, planning phases (current, migration, target).

Software architecture deals specifically with the internal structure of single software systems, or families of similar software systems (product lines). It is a popular subject in information systems research. Enterprise architecture deals with the business functions and processes and links these to the software and hardware systems that support them. It is less popular in IS research and concepts and approaches are often borrowed from software architecture. Other specializations in the domain of architecture exist. The companies that supported this research were primarily interested in enterprise architecture. During our research we have felt no need to explicitly distinguish the kind of IT-architecture we were dealing with.

Van Vliet (2000) places the software architecture phase in the software life cycle between the requirements engineering and design phases. During the architecture phase, the interests and concerns of all stakeholders are taken into account to come to a well-balanced solution.

The result of the architecture phase is a series of major design decisions that put constraints on the building process as well as the product delivered. These design decisions are represented in a description, which is often in the form of a series of diagrams. These diagrams give different views of the system, such as the decomposition of a system into major logical building blocks, or the mapping of software elements onto hardware elements.

1.3 What is Communication?

During our research we never felt the need to choose a formal definition of communication; we worked with the general understanding of the term. We often explained our research in this way: ‘we study high level design documents and want to see how we can get every stakeholder to receive the information he really needs’. Usually this would arouse a positive and understanding reaction. Evidently ‘giving someone the information he really needs’ is an important aspect of communication. While writing this thesis we find time to investigate the term further and see (in the next section) whether this leads to a clearer formulation of the scope of our research project.

The Merriam Webster On-line dictionary gives this definition of communication:
1 : an act or instance of transmitting

2 a : information communicated **b :** a verbal or written message
3 a : a process by which information is exchanged between individuals through a common system of symbols, signs, or behavior <the function of pheromones in insect *communication*>; *also* **:** exchange of information **b :** personal rapport <a lack of *communication* between old and young persons>
4 *plural* **a :** a system (as of telephones) for communicating **b :** a system of routes for moving troops, supplies, and vehicles **c :** personnel engaged in communicating
5 *plural but singular or plural in construction* **a :** a technique for expressing ideas effectively (as in speech) **b :** the technology of the transmission of information (as by print or telecommunication)

We see from this definition that communication encompasses a whole range of aspects. We will come back to this definition in the next section and continue with some scientific models that have been proposed to capture essentials of communication.

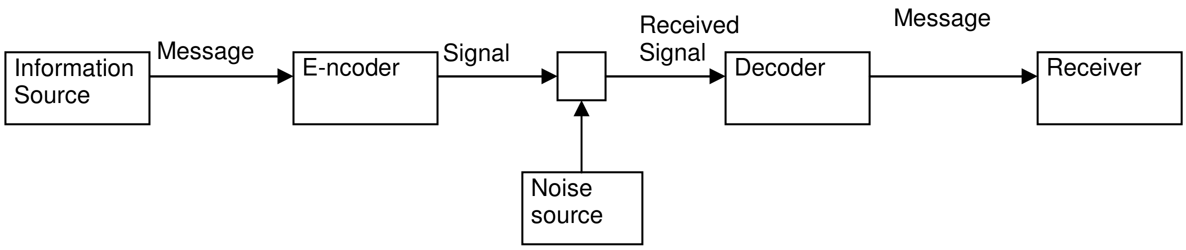


Figure 1-1 The Shannon-Weaver Model

The famous Shannon-Weaver Model (Underwood, 2003) proposes that all communication must include six elements: a source, an encoder, a message, a channel, a decoder and a receiver. It is an appealing, inspired by information technology, but simplified model, that does not take into account the context, the intentions and the choice of channel or medium. Matters such as the social context in which the message is transmitted, the assumptions made by source and receiver, their past experiences and so on are simply disregarded. When the message is in the channel, noise from other sources can distort the message. In the decoding of the message, semantic noise can arise. Examples are: distraction, differences in the use of the code, emphasising the wrong part of the message, attitude towards the sender, attitude towards the message. These criticisms of the Shannon-Weaver Model show some of the complexities that can be found in communication.

Many more theories exist about communication. Communication occurs in many different situations and not all situations have a similarity to communication of IT-architecture. We have selected an additional three theories that seem to point to aspects that, in our view, are relevant for the communication of IT architecture. We mention them here briefly and will try to apply them to IT-architecture in the next section.

Elaboration Likelihood Model, Cacioppo and Petty (1979). A communication is successful when a change of attitude occurs and this is most likely to happen when there is high elaboration. When people are motivated and able to think about the content of the message, elaboration is high. Elaboration involves cognitive processes such as evaluation, recall, critical judgment, and inferential judgment.

Network Theory and Analysis. Barnes (1954) is credited with coining the notion of social networks, an outflow of his study of a Norwegian island parish in the early 1950s. Network analysis (social network theory) claims that beliefs or behaviors are affected by the social structure of relationships around a person, group, or organization.

Speech Act. Speech act theory is built on the foundation laid by Wittgenstein and Austin. John Searle (1969) is most often associated with the theory. According to Searle, to understand language one must understand the speaker's intention. Since language is intentional behavior, it should be treated like a form of action. Thus Searle refers to statements as speech acts.

1.4 Communication of IT-architecture

In this section we want to bring the previously mentioned material on communication closer to IT-architecture.

If we apply the Merriam Webster definition of communication to IT-architecture, we see that it is an exchange of information by means of a common set of symbols and signs. Usually the IT-architecture information is about the future shape of a complex part of the information technology supporting a company. The information is recorded in plain text and in (lots of) diagrams. It is good to be aware of the fact that symbols and signs used in communication about IT need to be 'common' between author and readers otherwise the *readability* is at stake. Where acronyms or jargon is used, it is good to give explanations. Where applicable the terminology of a standard (from IEEE or other body) can be followed. The symbols used in diagrams should be explained. If an (elaborate) visual language is used, training to read the diagrams may be in place. The visual language of IT-architecture diagrams was a subject of our research, which you will find in chapter 2.

If we position the communication of IT-architecture in the Shannon-Weaver model, see Figure 1-1, we see the architect as the source. Encoding happens when the architect writes down his design for electronic or paper distribution, or speaks out in face to face meetings or presentations. The better the setup of architecture descriptions are (document template and diagram language) the more they are *easy to create*. The channels used are paper, computer screen or (sound in the) air. Decoding is the reading or hearing by the receivers. Here also goes, the better the document setup meets the reading predisposition of the receiver, the more they are *easy to understand*. Easy creating and easy understanding are the main aspects of chapter 7. The receivers of IT-architecture designs are very diverse. They may be system designers, reengineers, programmers, users, managers of user departments, managers of IT departments, auditors, and fellow architects. In this research we have focused on the paper or electronic distribution. We have not investigated the points mentioned by the critics of the Shannon-Weaver model: social context, assumptions made, past experiences and semantic noise. These could be subject of future investigations.

The three selected communication theories point to aspects that contribute to effective communication. The *Elaboration Likelihood Model* points to the fact that the architect must provide motivational arguments for his proposal early on (why is it important to read the design and think about it?). For instance the developers read the architectural design to decide on the possible building blocks of the code and to estimate whether they will have the desired behaviour. The architect should mention attention for this concern and the other relevant concerns in the introduction. Architectural descriptions are by nature documents that need thorough information processing. The reader must *elaborate* on the content to construct a mental model of the proposed architecture and to infer possible positive or negative effects seen from the perspective of his personal concerns. The *Network Theory and Analysis* model maintains that the architect preferably should be or become part of the social network of the stakeholders. This means that the architect should spend time with the stakeholders, listen to their concerns and build a relationship. Because the stakes are high and the architecture document often is the foundation for a major project, emotions can get involved. Trust in the architect as a person may be needed to go along his proposed path. The *Speech Act* model comes close to this by stressing that intentions of the architect should be clear to the stakeholders. An architect can do this by listing in the rationale all the concerns that have guided the design. If the stakeholders can identify with these concerns, that makes it easier for them to also identify with the design. There should be no more hidden goals or desired side effects.

The various closely related aspects of these three communication theories come into play in our research mainly in chapter 5.

In our research we did not touch on the social network aspect, as already mentioned with the Shannon-Weaver model. But we did touch upon the motivation (concerns) of the receivers (stakeholders) and the intentions of the architect. If we formulate the intentions of the architect as devising a solution that meets as much as possible the concerns of the stakeholders, we are at the heart of IEEE Std 1471 which played an important part in our research. See section 1.5 for an introduction of IEEE Std 1471.

The strategic importance, the diversity of the readers and the abstract, non-tangible content make the communication of IT-architecture quite a challenge and an interesting research subject.

The dealings of IT-architects with management can be placed in the broader context of domain experts communicating with management. Eppler (2004) has studied the communication problems between experts and managers, and has found, amongst others (page 15): "Summarizing these issues, we can conclude that experts struggle with three major issues when transferring their knowledge to managers: First, reducing or synthesizing their insights adequately, second, adapting these trimmed insights to the management context without distorting them, and third, presenting the compressed and adapted findings in a trust-building style and reacting adequately to management questions and feedback." We return to this in chapter 7.

In the concluding chapter (chapter 8) of this thesis we summarize our findings regarding the communication theories mentioned in this section.

1.5 IEEE Std 1471

Chapters 3, 4 and 5 of this research rely heavily on IEEE Std 1471 (IEEE 2000). That is why we are mentioning it here in the introduction and are giving you a glimpse of its content. Chapter 3 offers a more elaborate introduction of the standard and a picture of its conceptual model. See chapter 8 for a recap of our conclusions concerning IEEE Std 1471.

IEEE Std 1471 was defined by the IEEE Architecture Working Group in the period May 1996 to December 1999. There were 29 participants and 137 reviewers. The standard received a warm welcome in the architecture community, but is yet nevertheless not often applied.

The purpose of IEEE Std 1471 is ‘to facilitate the expression and communication of architectures and thereby lay a foundation for quality and cost gains through standardization of elements and practices for architectural description’ (p.1). The standard defines ‘useful terms, principles and guidelines for the consistent application of architectural precepts to systems throughout their life cycle’ (p.iii). It also provides guidance on the structure of architectural descriptions. The main concepts standardized are “architecture”, “architectural description”, “concern”, “stakeholder”, “viewpoint” and “view”. Architectural descriptions are segmented into views. A view addresses a related set of stakeholder concerns and is constructed in accordance with the specification that is laid down in a viewpoint. Together the views address all the concerns of the stakeholders.

IEEE Std 1471 gives definitions for the terms used (p. 3 and 4). The definition of *architecture* is already cited in section 1.2. An *architectural description* is a collection of products to document an architecture. *Concern*, though a key concept, is not defined. A system *stakeholder* is an individual, team, or organization (or classes thereof) with interests in, or concerns relative to, a system. A *viewpoint* is a specification of the conventions for constructing and using a view, a pattern or template from which to develop individual views by establishing the purposes and audience for a view and the techniques for its creation and analysis. A *view* is a representation of a whole system from the perspective of a related set of concerns.

The standard demands as a minimum that the following stakeholders are recognized: users, acquirers, developers and maintainers. At a minimum, the concerns identified should include: the purpose or missions of the system, the appropriateness of the system for use in fulfilling its missions, the feasibility of constructing the system, the risks of system development and operation to users, acquirers, and developers of the system, and maintainability, deployability, and evolvability of the system.

The fact that the standard is not applied much yet, aroused our interest in perceiving the *distance* between IEEE Std 1471 and current practice. To what extent is current practice in line with IEEE Std 1471? Does applying the standard require a new line of thinking from the architects?

IEEE Std 1471 was accepted in 2001 by ANSI as a joined standard. In 2006 ANSI/IEEE Std 1471 was adopted by ISO as a draft international standard ISO/IEC DIS 25961. It was announced that ISO and IEEE will undertake a revision to resolve the comments received in the ballot process. ISO/IEC JTC 1/SC 7/Workgroup 42 is aiming at producing the final standard as ISO/IEC 42010.

The interesting point with regard to the communication of IT-architecture is that the content of architecture descriptions is explicitly linked to the concerns of the stakeholders. The description should become much better accessible to the stakeholder. The focus of our research regarding IEEE Std 1471 therefore was: the segmentation into views that are related to stakeholder concerns.

1.6 Research Questions

This research into communication of IT-architecture has two goals: acquiring knowledge about the work of IT-architects and contributing to the work of IT-architects by means of guidelines. The overall research question is:

How can the practices of communication of IT-architecture be improved?

In the past years we have performed various research activities regarding communication of IT-architecture. In these activities we have encountered and sought answers to the following sub questions.

Q1 How can the readability of IT-architecture diagrams be improved?
Q2 How large is the gap between current architecture documents and IEEE Std 1471?
Q3 How to define IEEE Std 1471 viewpoints?
Q4 What are the concerns of the stakeholders and which concerns are not addressed by the current architecture documentation practice (for a given architecture department)?
Q5 What can we learn from a comparison of existing architecture frameworks regarding the communication of IT-architecture?
Q6 How can we describe Enterprise Architecture in a way that is easy to create (author) and easy to understand (reader)?

In section 1.10 ‘Introductions to publications’ we say more about how these questions came about and what motivated the research activities. See 1.10.1 for Q1, 1.10.2 for Q2, etc.

1.7 Research Setting

The research was performed in cooperation with three companies that have supported this research over varying periods. The companies are: the Belastingdienst, Ordina Panfox Public and ING. All three companies have a track record of applying architecture to manage IT investments and operations.

The Belastingdienst is a government office for collecting taxes and customs fares. The Belastingdienst has over 30.000 employees in offices all over the country of the Netherlands and a large central EDP department. They took part in this research in the year 2001 and contributed to chapter 2.

Ordina Panfox Public is an IT consultancy firm that specializes in government and municipal bodies. They have developed their own method for architecting the automation of large administrative operations. Ordina Panfox Public took part in this research from 2001 until 2003 and thus helped contribute to chapters 2, 3 and 4.

ING is a large international Dutch bank with over 110.000 employees and an EDP department of 10.000 employees, of which several hundreds IT architects. We cooperated with an architecture department formerly belonging to ING Barings and nowadays belonging to ING Wholesale. They maintain a database with information about more than 500 information systems and respond to inquiries about these information systems and write architecture reports with proposals of future sets of information systems to support the various business domains. ING took part from 2001 until 2004 and the outcome of our research done with them contributed to chapters 2 to 5.

Chapter 6 originated from a cooperation of professor Hans van Vliet, myself and an IBM architecture consultant Danny Greefhorst.

Chapter 7 is an academic intern production by professor Sjaak Brinkkemper, assistant professor Rik Bos and undersigned.

1.8 Research Methodology

This research is conducted in cooperation with practicing IT-architects. Their current practices and, where applicable, possible improvements, are the subject of this research. We investigated opinions of architects and stakeholders and proposed for some aspects improvements for practice. The research methodology that fits this approach best is called *action research* (AR)(Susman.and Evered, 1978, Baskerville, 1999; Davison et al., 2004). We mention this methodology in our reflection in hindsight on the validity of our research activities. We did not devise our research activities from the outset following strictly an established research methodology. Action research is applied where it is not possible or not viable to mimic a real life situation in a laboratory environment. In action research complex social processes are studied by introducing changes into these processes and observing the effects of these changes. Knowledge obtained through the use of this approach is difficult to validate in terms of the natural science view. Action research is valued for the relevant results it delivers and for the ability to take into account subjective experiences of human beings (as opposed to more classical positivists research which deals only with ‘hard’ tangible data).

In an action research setting the researcher is actively involved in the practice, with expected benefit for both researcher and practitioners. The researcher is on forehand not knowledgeable about the situation in practice. The knowledge obtained can be immediately applied to the situation at hand (there is not the sense of the detached observer) and does not necessarily need to be generally applicable. The research is a process linking theory and practice. We follow the interpretive stance of action research.

In action research five steps are defined: diagnosing, action planning, action taking, evaluation, specifying learning. During diagnosing a current practice is investigated together with the practitioners and problems to solve are formulated. In action planning and taking improvements are proposed and tried out. In action taking there is room to adjust the course of action in response to (unforeseen) developments. Evaluation is the judging of the improvements to see whether the problems are solved or not. Specifying learning is the academic part of it, the extension of knowledge (writing papers in our case). If the problems are not solved the circle recommences. These elements occur in varying degrees in most of the papers. Diagnosing is the main trait of chapters 3 and 5. Action planning, taking (proposing and trying improvements) and evaluation are described in chapters 2, 4 and 7. All the papers reflect specifying learning, but it was the main goal of chapter 6.

The diagnosing activities of real life situations which underlie chapters 3 and 5 can be seen as independent investigations that generated ideas. In that sense they are characteristic of so called illustrative *case study research* (Yin, 2003; Tellis, 1997). In case study research a small number of instances are investigated in depth. Data is collected and analyzed, which leads to a sharpened understanding or new hypotheses and possibly further research. We see chapter 4 as the new understanding or hypotheses promoted by chapter 3. In chapter 5 the investigation led to a proposal for changes in the documentation practice; these changes could be investigated further for their effect.

The observation techniques used were: questionnaire, semi-structured interviews, discussions and analysis of original documents.

1.9 Related work

In this section we mention some of the other research that is taking place in the realm of IT-architecture. IT-architecture is rather new as a research subject and is very open to different kinds of contributions.

- Smolander and Päivärinta (2002) have done research regarding the use of IEEE 1471. They examined the reasons for making architecture descriptions in practice. Interviews with various stakeholders of architecture in three companies showed that beside the traditional use as a starting point for system design, architecture documents serve to communicate, to negotiate and to capture knowledge. According to the authors, these other uses should lead to new viewpoints and new tool support. One of the recommendations of Smolander and Päivärinta is to do further research into "how communicative these descriptions appear to be in the varying contexts of their use". A recommendation we have taken at heart.
- Rikard Land (2003) has gained experience with "Applying the IEEE 1471-2000 Recommended Practice to a Software Integration Project". IEEE Std 1471 was used in a project where developers from different companies had to compile shared architectural views on three existing systems and on alternative concepts for one integrated system. The discussion was guided by a separate list of stakeholder concerns, which was extended or modified from time to time as the discussions revealed additional concerns. The concern concept of IEEE Std 1471 is also central in our research, see for instance the Concern Inquiry Tool in chapter 5.
- Nico Lassing (2002) in his Ph.D. thesis has worked out a method to use the architecture level to assess the modifiability of a system at a very early stage. For stakeholders whose main concern is modifiability, he proposes four *viewpoints* that address this concern. The viewpoint concept is also central in our research, but our approach is more generic.
- The Software Engineering Institute has for some years now been contributing to the development of the profession of software architects. Bass, Clements and Kazman (2003) give 'practical guidance on managing software architecture in a real software development organization' (p. xi). Subjects treated are among others, quality attributes, *documentation (views)*, analyzing (evaluating) architecture designs and product lines. Clements et al (2003) specifically deal with documenting software architecture. They offer many

reusable *viewpoints* to be used in practice. This thorough and extensive work shows the importance of architectural documentation. Our research adds to this: support for a situational approach to viewpoints, some additional enterprise architecture viewpoints and concepts for organizing large collections of architectural descriptions.

- Gerrit Muller is slowly, but steadily, building his Gaudi website (Muller, 2007). “The ambition of the Gaudí project is to make the art and emerging methodology of System architecture more accessible and to transfer this know-how and skills to a new generation of system architects.” Topics covered are, among others: what is an architect and how does he work, the CAFCR *framework*, architectural reasoning, composable architectures (product families). These experiences originate from the context of producing industrial, embedded software. Where Gerrit proposes one fixed framework, we adhere to a more flexible approach to architectural views. Our roots lie in software supporting business functions in large enterprises.
- The Archimate project (Lankhorst, 2005; Archimate, 2008) has produced a standardized language for describing enterprise architectures. The conceptual model consists of 26 entities and for each concept a visual representation is defined. 16 diagram types (viewpoints) are proposed as a basic set to work with the language, but many more could be constructed. Together this offers a powerful way to communicate IT-architectures. The Archimate project was an inspiration for us to develop a simplified language and diagram types for modelling the relation between enterprise functions and IT-solutions.
- In the GRAAL (2008) program continued research is done into the alignment of ICT architecture and business architecture. The program has produced amongst others a *framework to describe* architecture which we have taken into account in our comparison of existing frameworks. Current topics in GRAAL are: quality indicators, value based alignment, coordination processes and electronic intermediation services.
- The GRIFFIN (2008) project “develops notations, tools and associated methods to extract, represent and use architectural knowledge that currently is not documented or represented in the system.” This project is trying to cover up for missing information in the communication of IT-architecture and is very complementary to our research.
- The Open Group (2002) has developed TOGAF (The Open Group Architecture Framework), which has as its core the TOGAF-ADM (TOGAF Architecture Development Method) a step-by-step approach to developing enterprise architecture. The other main pillar is the Enterprise Continuum, a virtual repository of architecture assets, which includes the TOGAF Foundation Architecture, and the Integrated Information Infrastructure Reference Model. TOGAF is important for the communication of IT-architecture because it creates a common frame of reference between its many followers. TOGAF does not prescribe views. Architects working in a TOGAF ‘shop’ can find in our research tools and methods which can be of use in their work, for instance to create the needed architectural views. The emphasis of TOGAF is on the architectural process. Our emphasis is on the structure of the resulting architecture descriptions.
- Jaap Schekkerman is slowly, but steadily, building his IFEAD website (Schekkerman, 2008), the website of the Institute For Enterprise Architecture Developments. It offers links, standards, methods, tools, best practices, books, conferences and lots of other stuff. His Extended Enterprise Architecture Framework (EEAF) lists the relevant items to describe an enterprise and its information technology in a holistic way. Architects considering EEAF for their situation can use our proposed method for viewpoint design to evaluate EEAF for their situation.
- A conference series to watch is WICSA (2008), the Working IEEE/IFIP Conference on Software Architecture. “The mission of the WICSA conference series is to be the premier means of communication and advancement of research and practice in software

architecture, from both academia and industry, worldwide." From time to time there are presentations about architectural views in WICSA.

1.10 Introductions to publications

In this section we introduce the publications of this research. For each publication we briefly mention the project circumstances in which the idea for conducting such a research activity originated, we explicitly outline the motivation for the activity and we mention the main research question that was answered by the activity. These research questions are already listed together in section 1.6.

The first four publications represent work that is mainly my own. The co-authors contributed by pointing to reference material and by guiding us in learning to write scientific papers (instead of business reports or research logs). The idea for the fifth publication was worked out in discussions with Danny Greefhorst in which we both contributed equally. Together with Hans van Vliet we performed a major overhaul with an eye on international publication. As for the last paper, I elaborated on the groundwork laid by Sjaak Brinkkemper and Rik Bos, who also actively took part in processing received review comments.

1.10.1 Practical Guidelines for the Readability of IT-architecture Diagrams

Koning, H., Dormann, C., van Vliet, H., 2002. Practical Guidelines for the Readability of IT-architecture Diagrams, in: Proceedings ACM SIGDOC 2002, pp 90-99.

Out of our interest in visualization we started collecting guidelines for making IT-architecture diagrams in order to improve their readability. Based on our own experience as IT-architects we browsed through all sorts of sources in search for guidelines: books on documenting information systems, books on website design, and books on human perception and on psychology (Gestalt theory), articles about colour theory. We held conversations with architects with hands-on experience and noted their ideas. We browsed the Internet in search for papers and informative web pages. In the end we had collected 190 statements (directions 'do this', 'do that') on making diagrams in IT. These statements were presented by means of workshops to groups of practicing architects at the three companies that supported this research. After three months a vote was taken, in which the architects indicated for each statement whether they thought it was helpful for their work or not. We demanded 70% support. There were 32 statements that got less than 70% support and they were evicted from the list. See chapter 0 of this thesis for the resulting list of guidelines. We held the workshop several more times, also on a commercial basis for completely different groups.

Diagrams play an important role in the communication of IT-architecture. The value of diagrams in communication is very often expressed by the saying 'A Picture is Worth Ten Thousand Words'. Larkin and Simon (1987), in their classical paper, have argued that the value of diagrams lies in the ease of recognizing complex relationships between many elements. Diagrams contribute mainly by giving an overview of components and relations and by giving support in making inferences, see Gyselinck and Tardieu (1998). Diagrams speed up the processing of the information and they aid in remembering the information. We have compiled a collection of guidelines concerning the creation of diagrams. In the guidelines we find solutions to the following problems: Too much information in a diagram, which makes it crowded and unreadable; Forms, sizes and widths that speak a confusing language; Wrong signaling because of careless positioning or coloring; Connector mess and Misplaced use of text.

This paper addressed research question Q1 How can the readability of IT-architecture diagrams be improved?

1.10.2 Real-life IT architecture design reports and their relation to IEEE Std 1471 stakeholders and concerns

Koning, H., van Vliet, H., 2006a. Real-life IT architecture design reports and their relation to IEEE Std 1471 stakeholders and concerns, Automated Software Engineering (ASE) Journal, Volume 13, Issue 2, pp. 201-223.

In 2002 I was asked if I would temporarily be interested in doing a topic of more general interest and then return to diagrams. I settled for IEEE Std 1471. This standard seemed to offer good starting points scientifically and practically. The following investigation has been the source of inspiration for the method viewpoints design in 2003.

In 2000 an important contribution to the communication of IT-architectures was made by the publication of IEEE Std 1471 (IEEE, 2000). The standard was greeted with a warm welcome by the architecture community and the acceptance of its proposed terms has made talking about architecture easier. But the standard is not much followed when it comes to actual architecture descriptions. So we decided to investigate the gap between the standard and some current, non-IEEE Std 1471 architectural documents, and to see whether this gap could be bridged. Based on four existing architecture reports, we asked the authors thereof to indicate who, in their eyes, were the most important stakeholders of their document. Next we asked them to name the most important concerns of these stakeholders and to indicate where in the document the needed information was related to these concerns. We tried to see whether we could create stakeholder oriented views by rearranging the topics.
This paper addresses research question Q2 How large is the gap between current architecture documents and IEEE Std 1471?

1.10.3 A Method for Defining IEEE Std 1471 Viewpoints

Koning, H., van Vliet, H., 2006b. A Method for Defining IEEE Std 1471 Viewpoints, the Journal of Systems and Software, volume 79 no 1, pp. 120-131.

One of the conclusions of the document scans in 2002 was that compliancy with IEEE Std 1471 needs to be brought in the documentation process right from the start. So you need IEEE Std 1471 viewpoints. Since they are not readily available, you have to create them yourselves. We designed a method for the definition of IEEE Std 1471 viewpoints based on the analysis tools of the document scans. A viewpoint is a prescription for a view. To get better views you need better viewpoints, and that is the aim of the method. The method equips IT-architects to achieve IEEE Std 1471 compliancy and thereby create highly accessible architecture descriptions. In other words, it enables one to create architecture descriptions that communicate easily. The method consists of these steps: create stakeholder profiles, summarize architecture, relate summary to concerns from profiles and define viewpoints. To each step belong some modest tools (templates MS Word or MS Visio). To validate the method a round of discussion and try outs in practice was organized.

A lot of work went into evaluating the method. On two occasions the method was applied by a student or by student groups. With practicing architects we had workshop discussions and the method was applied in a situation where there was already documentation to define IEEE Std 1471 compliant viewpoints. The students created views with the viewpoints. Notes were taken in all the test sessions. We processed them carefully and distilled recommendations for the improvement of the method.
Despite the fact that no professional views have been written based on viewpoints which have been designed following this method, we are encouraged about the worth of the method because of the feedback we received from participating architects. Around the summer of 2003 we successfully guided an architect of Ordina in applying the method in his situation.

This paper addresses research question Q3 How to define IEEE Std 1471 viewpoints?

1.10.4 An Inquiry Tool for Stakeholder Concerns of Architectural Viewpoints: a Case Study at a large Financial Service Provider

Koning, H, Bos, R, Brinkkemper, S, 2006. An Inquiry Tool for Stakeholder Concerns of Architectural Viewpoints: a Case Study at a large Financial Service Provider, TEAR2006 workshop of the 10th IEEE International Enterprise Distributed Object Computing Conference (EDOC 2006), 16-20 October 2006, Hong Kong, China.

One of the feedback comments from the evaluation of the method viewpoints design was 'give us library viewpoints'. We picked up this feedback comment and endeavoured to create library viewpoints. In accordance with our method this meant that we had to establish the concerns of the stakeholders regarding the targeted type of architecture documents. To solicit the concerns of stakeholders we constructed an inquiry tool in the form of a list of interview questions.

Before the interviews started we have together with the contact person at ING predicted the concerns of the stakeholders and compiled IEEE Std 1471 viewpoints that describe the current practice of the architecture department at ING. We also held a review session with experienced practicing architects of the interview plans. This was very useful and some good points came out of these sessions. The interview was held eight times with stakeholders of the architecture department at ING.

This paper addresses research question Q4 What are the concerns of the stakeholders and which concerns are not addressed by the current architecture documentation practice (for a given architecture department)?

1.10.5 The many faces of architecture

Greefhorst, D, Koning, H, Van Vliet, H, 2006. The many faces of architectural descriptions, the journal of Information Systems Frontiers, volume 8 number 2, pp. 103-113.

Over the years all sorts of architectural frameworks have been defined. An architectural framework gives aid in managing a large collection of architectural documents by partitioning along dimensions. Persons who are new to the field, like junior architects, and who see so many diverging frameworks, ask themselves "what is going on here?". So we started collecting frameworks and started a comparison of the dimensions in them to see "what was going on there"; a very open research approach. We discovered nine 'base dimensions'. And we found out that one dimension is predominantly used.

This paper addresses research question Q5 What can we learn from a comparison of existing architecture frameworks regarding the communication of IT-architecture?

1.10.6 A Lightweight Method for the Modeling of Enterprise Architectures: Introduction and Usage Feedback

Koning, H, Bos, R, Brinkkemper, S, 2008. A Lightweight Method for the Modeling of Enterprise Architectures: Introduction and Usage Feedback, submitted for publication.

In 2005 I was informed of the work on enterprise architecture modeling (EAM). This subject was treated as part of a master course on Enterprise Architecture. I gave support by performing a needed revision of the course documents. For the resulting paper an inquiry was performed to evaluate EAM with the creators and with the recipients of EAM documents.

Enterprise architecture (EA) has established itself as a distinct type of architecture. EA describes the essential functioning of an organization and links that to the IT-support that is available. EA models give support in strategic decisions regarding IT. Despite its importance there is no agreed upon modeling method for EA and this hampers the communication of EA. Some methods exist but they are quite complicated. This means that a lot of time is needed to learn to apply the method and the results cannot be read without training in the language of the method. At the Utrecht University this need in the communication of EA was spotted and an easy, lightweight method for the modeling of EA was introduced. We gave help in documenting this method, extending it with a new diagram type and writing a scientific paper about it.

This paper addresses research question Q6 How can we describe Enterprise Architecture in a way that is easy to create (author) and easy to understand (reader)?

1.11 References

Archimate, 2008. Website (Dutch) with project results: http://www.archimate.org/en.

Barnes, J. (1954). Class and Committees in a Norwegian Island Parish. Human Relations, 7, 39-58.

Baskerville, R. (1999). Investigating Information Systems with Action Research, Communications of the Association for Information Systems, 2(19): 1- 31.

Bass, L, Clements, P, Kazman, R (2003). Software Architecture in Practice, Addison-Wesley.

Cacioppo, J.T. & Petty, R.E. (1979). Effects of message repetition and position on cognitive response, recall and persuasion. Journal of Personality and Social Psychology, 27, 97-109.

Clements, P., F. Bachman, L. Bass, D. Garlan, J. Ivers, R. Little, R. Nord, and J. Stafford, (2003). Documenting Software Architectures: Views and Beyond¸ Addison-Wesley, Boston.

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

Eppler, M.J., (2004) Knowledge Communication Problems between Experts and Managers. An Analysis of Knowledge Transfer in Decision Processes, ICA Working Paper #1/2004, University of Lugano, Lugano

GRAAL, 2008. Research program, website http://graal.ewi.utwente.nl/.

Greefhorst, D, Koning, H, Van Vliet, H, 2006. The many faces of architectural descriptions, the journal of Information Systems Frontiers, volume 8 number 2, pp. 103-113.

GRIFFIN, 2008. Research program, website http://griffin.cs.vu.nl/.

Gyselinck, V., Tardieu, H., 1998. The Role of Illustrations in Text Comprehension: What, When, for Whom, and Why?. In: van Oostendorp, H, Goldman, S. R., (Eds.) The Construction of Mental Representations During Reading, Lea, ISBN 0-8058-2429-4

IEEE, 2000. IEEE recommended practice for architecture description, IEEE Standard 1471.

A.S. Klusener, R. Lammel, C. Verhoef (2005) "Architectural modifications to deployed software", Science of Computer Programming 54 (2005) 143-211

Koning, H., Dormann, C., van Vliet, H., 2002. Practical Guidelines for the Readability of IT-architecture Diagrams, in: Proceedings ACM SIGDOC 2002, pp 90-99.

Koning, H., van Vliet, H., 2006a. Real-life IT architecture design reports and their relation to IEEE Std 1471 stakeholders and concerns, Automated Software Engineering (ASE) Journal, Volume 13, Issue 2, pp. 201-223.

Koning, H, van Vliet, H, 2006b. A Method for Defining IEEE Std 1471 Viewpoints, the Journal of Systems and Software, volume 79 no 1, pp. 120-131.

Koning, H, Bos, R, Brinkkemper, S, 2006. An Inquiry Tool for Stakeholder Concerns of Architectural Viewpoints: a Case Study at a large Financial Service Provider, TEAR2006 workshop of the 10th IEEE International Enterprise Distributed Object Computing Conference (EDOC 2006), 16-20 October 2006, in Hong Kong, China

Koning, H, Bos, R, Brinkkemper, S, 2008. A Lightweight Method for the Modeling of Enterprise Architectures: Introduction and Usage Feedback, submitted for publication.

Land, R (2003). Applying the IEEE 1471-2000 Recommended Practice to a Software Integration Project, in proceedings SERP'03 (International Conference on Software Engineering Research and Practice), CSREA Press, Las Vegas, Nevada.

Lankhorst, M. eds (2005). Enterprise Architecture at Work, Springer-Verlag Berlin.
Larkin, Jill. H.; Simon, Herbert A. (1987). Why a Diagram is (Sometimes) Worth Ten Thousand Words, Cognitive Science 11, 65-99
Lassing, N, 2002. Architecture-level Modifiability Analysis. Doctoral thesis, Vrije Universiteit, Amsterdam.
Muller, G, 2007. The Gaudy project. Website at http://www.gaudisite.nl.
Parnas, D.L. & Clements, P.C., 1986. A Rational Design Process: How and Why to Fake It., IEEE Transactions on Software Engineering, 12, 2, 251-257.
The Open Group (2002), TOGAF "Enterprise Edition" Version 8.1. http://www.opengroup.org/architecture/togaf8-doc/arch/
Rüping, A, 2003. Agile Documentation, A Pattern Guide to Producing Lightweight Documents for Software Projects. Wiley, Chichester, England. ISBN 0-470-85617-3.
Schekkerman, J, 2004. How to survive in the jungle of Enterprise Architecture Frameworks, Trafford, Victoria, Canada.
Schekkerman, J, 2008. The website of the Institute For Enterprise Architecture Developments (IFEAD) http://www.enterprise-architecture.info
Smolander, K. and T. Päivärinta, 2002. Practical Rationale for Describing Software Architecture, Beyond Programming-in-The-Large, proceedings of 3rd Working IEEE/IFIP Conference on Software Architecture (WICSA3), August 25-30, 2002, Montreal, Québec, Canada, Jan Bosch et.al., ISBN 1-4020-7176-0, page 113-125.
Searle, J.R. (1969). Speech Acts: an essay in the philosophy of language. Cambridge: University Press.
Software Engineering Institute, 2008. How Do You Define Software Architecture? http://www.sei.cmu.edu/architecture/definitions.html, accessed July 12, 2008.
Susman, G.L.and Evered, R.D, 1978. An assessment of the scientific merits of action research. Administrative Sciences Quarterly, 23, 582–603.
Tellis, W., 1997. Introduction to case study [68 paragraphs]. The Qualitative Report [On-line serial], 3(2). Available: http://www.nova.edu/ssss/QR/QR3-2/tellis1.html
Underwood, M, 2003. The Shannon-Weaver Model, webpage at http://www.cultsock.ndirect.co.uk/MUHome/cshtml/introductory/sw.html.
Van Vliet, H, 2000. Software Engineering: Principles and Practice, John Wiley & Sons, Chichester.
WICSA, 2008. The Working IEEE/IFIP Conference on Software Architecture. http://www.softwarearchitectureportal.org/WICSA/conferences/
Yin, Robert K, 2003. Case study research: design and methods, 3d ed., Sage Publications, Thousand Oaks.

2 Practical Guidelines for the Readability of IT-architecture Diagrams

Henk Koning Claire Dormann Hans van Vliet
Faculty of Science
Vrije Universiteit, Amsterdam
De Boelelaan 1081a,
1081 HV Amsterdam, The Netherlands
henk@cs.vu.nl claire@cs.vu.nl hans@cs.vu.nl

ABSTRACT

This paper presents the work done to establish guidelines for the creation of readable IT-architecture diagrams and gives some examples of guidelines and some examples of improved diagrams. These guidelines are meant to assist practicing IT-architects in preparing the diagrams to communicate their architectures to the various stakeholders. Diagramming has always been important in information technology (IT), but the recent interest in IT-architecture, the widespread use of software and developments in electronic communication, make it necessary to again look at the 'art of making diagrams' for this particular class and its users. The guidelines indicate how various visual attributes, like hierarchy, layout, color, form, graphics, etc. can contribute to the readability of IT-architecture diagrams. The emphasis is on the outward appearance of diagrams. Some additional support is given for the thinking/reasoning processes while designing or using a set of diagrams and an attempt is made to arrive at a rationale of these guidelines. An evaluation process has been performed with three groups of practicing IT-architects. The outcome of this evaluation is presented. This work is part of a more comprehensive research project on "Visualisation of IT-architecture".

GENERAL TERMS

Documentation, Design, Human Factors.

KEYWORDS

Visualization, Architecture, Diagrams, Guidelines, Readability, Hierarchy, Layout, Color, Text, Form, Size, Width, Graphics.

2.1 Introduction

We present work done to establish guidelines for the creation of readable IT-architecture diagrams and give some example guidelines and some examples of improved diagrams. We do not come up with new insights into the use of diagramming and diagrammatic reasoning in general. Rather, we try to bring general research findings to the domain of information technology (IT).

This work is part of a more comprehensive study about "Visualisation of IT-architecture".

In this introduction we say a few words about IT-architecture in relation to diagramming, we mention some current developments therein, and indicate the current status of our work in establishing guidelines. In section 2.2 we give a short description of the approach we took and we discuss the evaluation process. In section 2.3 we dig into the rationale of the proposed guidelines. In section 2.4 a condensed subset of the guidelines is presented and illustrated. In section 2.5 we give some support for the design and use of diagrams. We end in section 2.6 with conclusions and future work.

To view or print in color this document can be downloaded from the website of this research[2].

2.1.1 IT-architecture

Computer science has a long history of creating and using diagrams. Flowcharts, functional decomposition diagrams, input/output schemas are examples of such diagrams. See Martin [19] for the state of the art in software diagramming before the general introduction of the personal computer. Methods for analysis and design come with a drawing standard. A recent upsurge is UML for OO design, see Fowler [8]. A well-trained IT-specialist naturally draws good diagrams, doesn't he?

Software architecture is a relatively new branch within software engineering. The recent IEEE Standard 1471 [13] defines it as "Architecture is 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". Van Vliet [26] places the architecture phase in the software life cycle between the requirements engineering and design phases. During the architecture phase, the interests and concerns of all stakeholders are taken into account to come to a well-balanced solution.

The result of the architecture phase is a series of major design decisions that put constraints on the building process as well as the product delivered. These design decisions are represented in a description, which is often in the form of a series of diagrams. These diagrams give different views of the system, such as the decomposition of a system into major logical building blocks, or the mapping of software elements onto hardware elements. Well-known diagram-based models for describing software architectures were proposed by Kruchten [16] and Soni et. all. [17]. Boar [2] describes in detail a set of IT-architecture diagrams and gives advice on managing architecture on a companywide scale. See Koning [15] for an annotated set of IT-architecture diagrams found on the Internet.

Although the focus is on IT-architecture many guidelines in this research apply to diagrams in related areas as well. As an example of that kind of diagramming can serve the task models in user interface design in Welie [28].

2.1.2 Current Developments

So why look again at diagramming for IT? Several factors make it interesting to look at diagram representations of IT-architecture. New, less-technical roles emerge in the field of information technology, which are filled by people with no technical background. These non-technical stakeholders play decisive roles in the development, assessment and use of IT-architectures. Another development is that society is becoming more geared to visual communication and there is a demand for more appealing and better visualisations. Much research is done in developing new graphical metaphors. Spence [25] gives a state of the art overview. Next is the widespread availability of drawing software and graphic libraries. These makes 'old style' pencil and template drawings look outdated. A last reason to be mentioned is, that IT-architecture diagrams seem to be less formal than design-diagrams and new (fuzzy) rules must be developed for dealing with them.

Hofmeister [11] and Brown [5] have contributed to a debate about the use of UML for architecture diagrams. The adoption of the IEEE 1471 standard has spurred interest in creating specific viewpoints, based on (visual) models. The question 'how to represent software architecture?' is inspiring SEI [22] and SIGDOC [23] workshops. The outcome of our research can be of value for these developments.

[2] http://www.cs.uu.nl/people/koningh

2.1.3 Communicating architecture

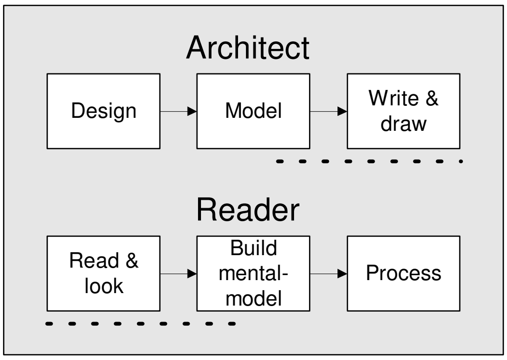


Figure 2-1 Steps in communicating architecture

For the sake of positioning the developed guidelines, the communication process can be divided in these broad general steps on the part of the author/architect:

- Designing the architecture
- Modeling the specification of the architecture for the communication, and
- Preparing the text and diagrams.

And on the part of the reader:

- Reading the text and diagrams
- Making a mental model of the architecture
- Processing the received information to validate, to accept or reject, to project consequences, etc.

The design step consists of understanding and structuring the needed features, understanding new technology, outlining various solutions, taking major design decisions, etc. It is a goal oriented, free format, creative thinking process. To get abreast of new concepts and technologies books, journals, conferences and seminars support the architect.

In the modeling step parts of the abstract architecture are made concrete. Patterns can be used. Formal semantic models like UML can be used. Company standards and culture can influence the partitioning.

The final step for the architect is preparing the actual deliverables.

In a design project these steps are not clearly divided. Actually there is a constant switching between free format thinking, modeling and drafting. The accent gradually shifts from the first step to the third.

Readers can be end-users, managers, other architects, developers, evaluators, etc. Comprehending the architecture starts with reading the documents. While reading, a mental model is created of the architecture, and the information is rationally and emotionally processed. Here also there is no sharp distinction between the steps, but the focus gradually shifts from mere reading/looking to realizing the consequences. When readers start to discuss with designers both ends meet.

The aim of the guidelines is to strengthen the transition from designer to reader. The scope of the readability guidelines is indicated in Figure 2-1 by the dotted lines: what can the architect do at the level of preparing the diagrams, to support the reader in quickly and correctly extracting the information? The focus is on the outward appearance of diagrams.

2.2 Activities and Evaluation Process

2.2.1 Activities

We started this research phase by establishing whether comparable work already had been done in this field. As far as we know this is not the case. So we decided to take a intuitive, broad-brush approach and establish a (rather informal) starting point.

Then we identified possible knowledge domains from which guidelines could be extracted. We used Ware [27] as a source on human perception, Horton [12] on technical documentation, Harst [10] on user interface design, Borchers [3] on website design. Other areas to be mentioned are: psychology (gestalt), diagrammic reasoning, information visualization, IT-modeling standards (UML) and icon libraries. Especially Horton [12] proved very useful.

In discussions with IT-architects of the participating companies heuristic rules (personal favorites) were captured. The guidelines were extracted based on our intuitive estimate of their possible value, based on our own experience as software developer and IT-architect.

In case of doubt about the usefulness of a guideline it was incorporated, on the premise that architects can for themselves decide what to use or not.

This resulted in a document with about 200 do's and don'ts, see Koning [14]. About this time we started rereading the same things again in other sources, which signaled to us that we were reaching a saturation point.

The result was presented to three groups of five to seven IT-architects of the participating companies (an international bank, a software house and a government institution). In the workshops that were held, the guidelines were applied as an exercise on existing diagrams. In the discussion of the guidelines in section 2.4, we show some real-life examples of IT-architecture diagrams that were used and improved in these workshops. The architects were asked to keep the guidelines at hand and apply them in their work as it suited them. They would be asked to give their personal estimate of the value after several months in the form of a questionnaire.

To strengthen the understanding of the guidelines a document was prepared in this period with examples of architecture diagrams that can be found on the Internet. Each diagram was commented on with respect to the various visual attributes. These example diagrams were distributed to the architects and can be found on the website of this research.

We concluded with a questionnaire.

2.2.2 Evaluation process

To keep it simple, in view of the large number of guidelines and the broad brush approach, the questionnaire consisted of one simple question for each guideline: keep or discard? It was possible to add a comment to this choice.

IT-architectects are an independent minded and busy bunch of people and they needed some urging to go this last step with us. Twelve architects responded to the questionnaire. Their responses varied from 'discard these 85 guidelines' to 'just keep them all and I'll gladly use them as a checklist'. In the end only five guidelines didn't receive 50% support to keep (they had a majority vote for 'discard'). We feel in this case it is appropriate to give more weight to the negative responses. If we draw the line on '70% support needed to stay in' another 27 guidelines fall from the table. In Table 2-1 this is summarized. The overall support column is calculated as % 'keep' of total votes for all guidelines for this visual attribute.

Table 2-1 Evaluation summary

Visual attribute	Number of guide-lines	Drop	Doubt-full	% Support
Hierarchy / focus	14	-	3	0,86
Form / size / width	8	1	1	0,86
Layout	29	-	4	0,86
Color	42	3	6	0,83
Connectors	16	-	3	0,86
Use of text	11	1	-	0,89
Graphics & icons	15	-	3	0,82
Context & design	45	-	5	0,88
Use in Report	10	-	2	0,88
Totals	*190*	*5*	*27*	

Some of the comments made by the respondents were: not a guideline but an explanation; not generally applicable (too specific); redundant (already in another guideline); too theoretical (mostly on perception). In sections 2.4 and 2.5 we give some examples of rejected or considered doubtful guidelines.

Our original document also contained a summary of 'Gestalt' rules, which were given as background in perception theory. Based on the voting these must also be considered doubtful. We expected (hoped to unleash) discussion from the participating architects and a rejection of a substantial number of guidelines. The discussion did not take place much, however, that was 'too far away' from their practical situation. The amount of rejected or doubted on guidelines supports the credibility of the (remaining) set of guidelines. Overall the architects appreciate the effort we have done to provide them with practical guidelines.

So the credibility of these guidelines is based on the quality of the sources, on our personal judgments and on taking votes from a small group of practicing architects. Our claim is not that these guidelines are proven correct. Our claim is that these guidelines are considered valuable by practicing architects.

2.3 Rationale of the Readability Guidelines

In this section we want to give some pointers to 'explain' the guidelines to IT-architects and to further motivate their use. In this we go only one step into the unknown. Further explanation for the interested reader can be found in the reference material.

The rationale behind the readability guidelines as a whole centers around four points: Human Perception, Appeal, Building the Mental Model and Support while Processing which are outlined in the next paragraphs.

2.3.1 Human Perception

First and foremost reading and looking are functions of human perception, which is in itself an amazing and very complex process. As Bertin [1] pointed out all the visual attributes play their role in helping the human brain in recognizing and grouping objects in the diagrams.

The guidelines also point to limits in human perception: the maximum numbers of objects, colors, forms, sizes, etc, a user can comfortably handle. The viewer needs a certain minimal differentiation between types of objects, a certain minimum space between objects, etc.

2.3.2 Appeal

Another major concern around the use of diagrams in IT-architecting is that diagrams must be appealing. Is the diagram attractive? Does the first impression stimulate the reader to dive into it and take in the information contained in the diagram? In modern day world most people are overloaded with information. If it doesn't look nice, it is easily put aside. The competition is on.

2.3.3 Building the Mental Model

When someone is reading the text and viewing the diagrams of an architecture description, he is building a mental model of the architecture. These elements of the mental model may be explicitly defined or tacitly implied. While viewing a diagram, a constant process of constructing possible objects, relations and attributes, evaluating the plausibility of the constructions, and affirming or rejecting them, is going on. This process is based on visual attributes present in the diagram. Information from already seen diagrams is taken into account, as is information from previous experiences (prior knowledge).

Our guidelines try to prevent as much as possible the construction of incorrect elements in the mental model and to promote as much as possible the construction of correct elements. Visual attributes of diagrams are what Norman [21] calls 'affordances' that provide strong clues as to their meaning. Creating diagrams without taking into account all the visual attributes that lead the viewer, results in meaningless variation and confusion. The value of diagrams in communication very often expressed by the saying 'A Diagram is Worth Ten Thousand Words'. Larkin and Simon [17], in their classical paper, have argued that the value of diagrams lies in the ease of recognizing complex relationships between many elements.

2.3.4 Support while Processing

Diagrams not only transfer information to build up a mental model, according to Narayanan [20], they also assist in processing the information. Viewing a diagram of an architecture helps keeping part of the model conscious in mind and it inspires and corrects thinking. It inspires because you can combine objects and/or attributes that you see in the diagram and construct alternative diagrams. It corrects because you 'see' more easily what is possible and what isn't.

For reasoning you need a starting point. In natural language, a starting point is created by expressions like "lets presume …, what consequences might that have?". The closer a diagram comes to your ideal starting point for a line of reasoning, the better you can think. If you have to impose additional mental constructs on a diagram to create your starting point, you have less brain resources for thinking and the starting point is weaker and less inspiring.

2.4 Guidelines Concerning Visual Attributes

Our list of guidelines concerning visual attributes has the following sections: hierarchy/layers, forms/size/width, layout, color, connectors, text, and graphics & icons. For each section we give some examples of the guidelines and some of the rationale.

2.4.1 Hierarchy

A diagram may contain more elements than can be consciously viewed at in one glance. That need not be a problem, provided the viewer is given enough clues to easily perceive the main message, and separate this from any additional content. See Table 2-2 for examples of guidelines in this respect.

Table 2-2 Hierarchy Guidelines

Creating a clear hierarchy in complex diagrams
1. Design a complex diagram so it can be read in 30 seconds, in 3 minutes, and in 30 minutes. A diagram (graphics) must immediately and automatically make the most important point, then present secondary points, and with study reveal details. A diagram must organize information into a clear visual hierarchy. 2. Different visual attributes can play a role in indicating the hierarchy: size (bigger), colors (brighter), lines (thicker), pattern (emphasize the main path, shorter lines), etc. 3. Avoid more than three distinct visual levels. 4. Use primary colors only for objects that need immediate action.

If you don't add hierarchy to a complex diagram, you require the viewer to derive the hierarchy (levels of importance) while viewing and to keep that information in his head, while processing the diagram further. This incurs extra work for the viewer, possibly irritation (up to quitting altogether) and possibly wrong conclusions. So a complex diagram without hierarchy is not appealing and has a high chance of not being properly understood. See Figure 2-2 for an example of a diagram with two visual hierarchical levels.

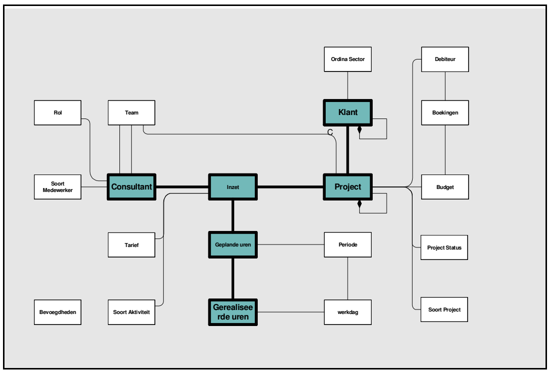


Figure 2-2 A diagram with two visual hierarchical levels

An example of a considered doubtful guideline in this category is "to create a sense of urgency in a diagram use simple graphics or 'unstable' graphics (like something almost falling to the ground)".

2.4.2 Forms of Objects, Size, Width

Forms used in IT-architecture diagrams generally fall into two categories: first there are the formal shapes like boxes, diamonds, data stores. Second there are the freestyle small graphical images like icons and small clipart. The use of graphics and icons is rather new and is treated in more detail in section 2.4.7. See Table 2-3 for examples of guidelines in this respect.

Table 2-3 Forms of objects Guidelines

Forms of objects
1. Be clear about what your forms mean. Are all boxes equal? And do they then mean the same too? If not, provide annotation. Be consistent in the use of forms in a set of diagrams. 2. Try to match the outward forms of objects to the intrinsic properties of the objects. 3. Use more detailed, realistic, three-dimensional symbols for concrete and tangible objects and use simple, geometric shapes for abstract concepts. 4. Don't use more than 6 different forms in one diagram.

An example of a rejected guideline in this category is "high level -> low level: more shading in the shapes".

Size and width are important visual attributes. Many people underestimate how strong the size-signal is in real life. A taller person is easily seen as more important. A bigger car, house, etc. induce similar connotations. Different sizes in one diagram would 'normally' mean differences in importance. See Table 2-4 for examples of guidelines in this respect.

Table 2-4 Size and Width Guidelines

Size and Width
1. See to it that similar objects have equal sizes/widths. Only deviate in size and width if you want to signal something. 2. Avoid resizing objects because of short or long texts. For long texts the alternative is to put a short label in the object and the text in an annotation (insert in the diagram) or in a textbox in the margin of the diagram. 3. Avoid making objects smaller to have more information in one diagram, or making objects bigger to fill up a diagram that otherwise looks too empty.

Different sizes of the same object in different diagrams normally means the object has grown or shrunk. See Figure 2-3 and Figure 2-4 for an example.

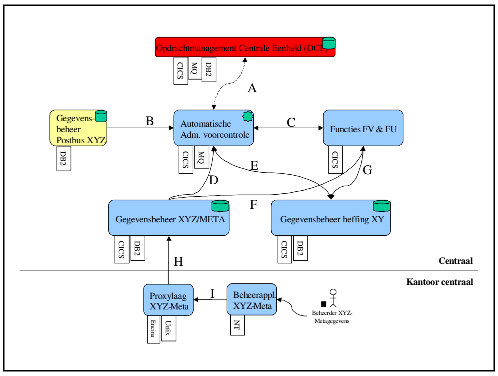


Figure 2-3 A diagram from a workshop participant

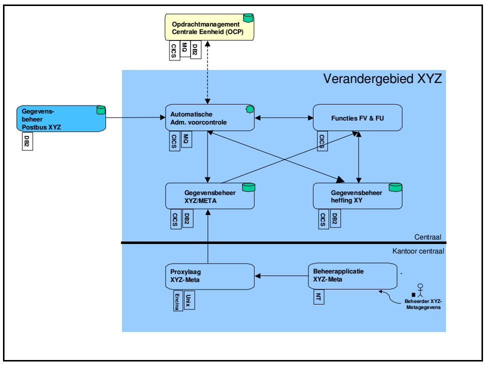


Figure 2-4 The same diagram as Figure 2-3 after applying guidelines about Size and Layout

In Figure 2-3 the objects have different sizes, but this has no meaning, so the 'size message' has to be suppressed in interpreting the diagram. The vertical position of the objects is meaningful, but the horizontal position is vague. This means the horizontal positions all have to be checked and interrelated to see whether there is some significance. In Figure 2-4 the sizes of the objects have been made equal and the objects have been positioned on a grid. This means much less parsing of the diagram has to be performed and one can start reasoning about the meaning immediately.

2.4.3 Layout

Putting the objects in a diagram in a pattern that is easily recognizable and fitting to the underlying message, is a great aid to the viewer of the diagram. It very much helps in discerning and remembering which objects there are and which relationships are relevant to consider. A clear pattern makes it easy for the eye to come back to objects that were already perceived, and thus supports processing.

Layout aspects of a diagram include: basic pattern, horizontal and vertical alignment, above/before positioning, symmetry, distance of objects from the center and from other objects, distribution of white space. A basic pattern makes clear to the viewer what strategy is being followed in positioning objects and what meaning can be derived from the position of an object. For instance: in a workflow diagram the activities might be positioned from left to right in the order of execution and having the same vertical position can mean being executed in the same stage of process.

Clearing up messy diagrams often starts with improving the layout, so it is one of the most important visual attributes. A good layout is perceived instantly and almost unconsciously. An unclear layout keeps nagging and hinders perceiving the more detailed information.

Providing enough, but not too much, white space makes diagrams elegant. White space gives room to envision alterations or additions, and in that way (again) supports reasoning about the diagram.

Figure 2-4 is an example of a diagram in which attention was paid to choosing a basic pattern and proper horizontal and vertical outlining of objects.

Table 2-5 Layout Guidelines

Layout
1. Choose a clear, recognizable positioning pattern for the objects in a diagram. Familiar layout patterns are: chain, grid, tree, web.
2. In positioning objects 'natural positions' should be preferred, e.g. central horizontal and vertical axis, secondary axis on ¼ and ¾ or 1/3 and 2/3, etc.
3. Objects should be positioned on horizontal and vertical lines, e.g. not positioning them so should be meaningful.
4. Take into account the 'natural flow' from left to right, and from top to bottom. Make other flow directions clearly recognizable.
5. Provide enough white space. A crowded diagram looks obtrusive.
6. Use white space to distinguish objects, borders, groups, etc.
7. In a set of diagrams similar objects should have a similar position.

Figure 2-5 is another example of a diagram with a clear layout pattern. The main page is divided in areas that convey the main structure of the architecture. In this case it shows how an application on a client workstation in the center of the left main area is serviced by applications on several local LAN servers and some remote midrange and mainframe machines. It is realistic, in that it deals with real machines and locations. It is conceptual in that some of the represented machines are actually groups of equivalent machines, and in that the whole left main area is in reality occurring three times at different locations. The operational division between LAN-hardware and midrange/mainframe is made clear by the extra white space area.

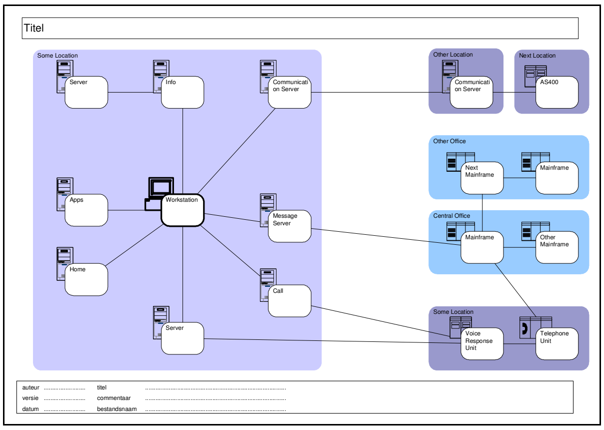


Figure 2-5 An example of a clear layout pattern

2.4.4 Color

Color is a very strong visual signal, so it is worthwhile paying attention to it. It is a visual attribute that is strongly influenced by 'prior knowledge', like cultural values, fashion colors in clothing or magazines, or company colors. Color is also rather new in computer documentation. Even nowadays not all practicing IT-architects have a color printer at hand. Due to the newness of the subject, the guidelines contain an element of 'color education', which may over time become less relevant.

Additional meanings can be easily (temporarily) attached to a certain color. Using a distinct color in a diagram for an object with a particular attribute, can program the meaning of that color for the rest of the documentation. Color can enlarge the appeal of the diagram.

Table 2-6 Color Guidelines

Color
1. Use color. The competition for the attention of the viewer is on!
2. Use color with restraint. Problems with (too much) color: wrong prior associations, distraction, tiresome, less legible, fuzzy, unreliable.
3. Don't use more than six colors in one diagram.
4. Color is especially useful to categorize objects, i.e. to group objects where other means (alignment, positioning) are not possible.
5. Use vivid colors only for strong signaling.
6. A safe rule is: choose light, non primary

colors with hues from over the whole color wheel.
7. If possible, follow the company colors.
8. Western viewers tend to prefer colors in the following order: blue, red, green, purple, orange and yellow.
9. To be recognized on screen colors need to be further apart than to be recognized on paper.
10. To avoid problems with colorblindness or with printing in black and white: use colors with different levels of brightness/lightness.
11. Let proximity in color parallel proximity in meaning (this extends over diagram borders).

The ‘fuzziness’ in the meaning of colors is reflected in some of the guidelines, which state ‘do something like …’. The combination of these guidelines can be used to setup a color scheme for a set of diagrams. Chijiiwa [6] gives many examples of color combinations and atmosphere.

In Figure 2-3 the color ‘vivid red’ was meant to indicate that the upper object is outside the scope of the system. The designer did not realize that the color red drew all attention to the object.

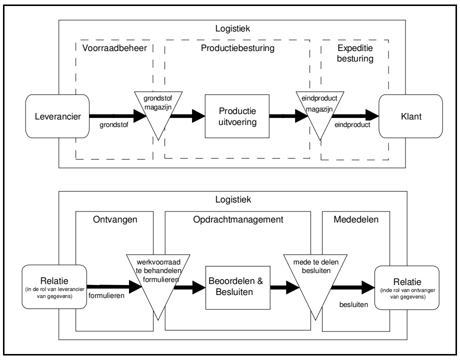


Figure 2-6 Another diagram from a workshop participant

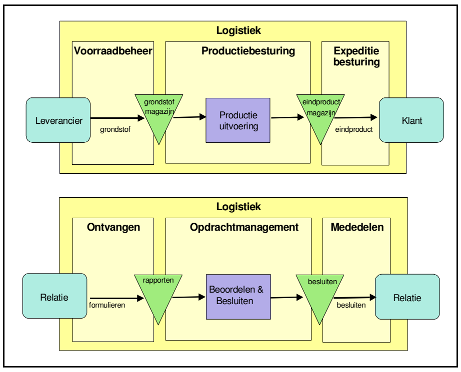


Figure 2-7 The same diagram as Figure 2-6 with colors

In comparing Figure 2-6 and Figure 2-7 you can see what a big difference adding colors can make. Figure 2-7 is much more appealing, the objects are more easily perceived against their background, and the semantic distance between the objects and the comparted areas is immediately clear.

An example of a rejected guideline in this category is “There are no ugly colors, or ugly color combinations. There are (combinations of) colors we are not used to … “

2.4.5 Connectors

Connectors are quite specific to IT architecture diagrams, which are all about objects and relations between objects. Not all architecture diagrams contain many connectors. A diagram showing how functionality is divided in three main areas may even do without any connectors. Process diagrams always contain connectors. It seems that the more precise the diagram is, the more design oriented, the more connectors come into play.

When connectors come into play, in number equaling or surpassing the number of other objects, they soon become problematic. Common problems are that connectors make the diagram look messy or overcrowded, and that connectors are not easy to follow. The usual solution is to try to find a better positioning of objects to make the connectors look better. This is not easy. Some designers give up and simply accept the mess, but this makes diagrams not appealing and more difficult to comprehend.

Some software tools optimize the flow of connectors. The problem with these tools is that the resulting layout totally ignores the already established, meaningful patterns in the diagram.

Table 2-7 Connector Guidelines

Connectors
1. Let the lines of connectors overlap, as long as this does not lead to unclearness or ambiguity. 2. Avoid unnecessary bends in connecting lines. 3. Avoid a too great emphasis on line-ends, like arrowheads. If possible, leave the arrowheads out all together. 4. Rounding of bends gives a more natural impression of flow.

5. Many close parallel lines are difficult to follow individually. A possible solution is to maintain different distances between (sets of) lines.
6. If parallel lines bend together, keep equal distances before and after bend.

Connectors that are easy to follow by the human eye give support to reasoning about the relationships between the objects. The guidelines about connectors give some ideas that make life with connectors more bearable. You can look at Figure 2-8 in section 2.4.7 and try to recognize some of the guidelines for connectors.

An example of a considered doubtful guideline in this category is “Give connectors a different line-width from other objects on the diagram. Smaller? Bigger?”

2.4.6 Use of Text

Text and graphics are friends and not enemies. Combine the power of 2. Text can be very strong in suggesting the proper interpretations and associations and in stimulating thinking. The guidelines on the use of text try to stimulate the architect to be diligent in adding proper titles, subscripts and annotations. They do matter. You can’t expect people to remember everything you said about this diagram in your (wonderful) real-life presentation. Text is important to speed up the building up of the proper mental model and to create a good starting point for a line of reasoning.

Table 2-8 Text in diagrams Guidelines

Text in diagrams
1. Write in active voice, use action words, use examples, tell how things look/sound/feel/smell, use concrete words that can be memorized verbally and visually. 2. If space does not permit a complete label, place a short label in or near the shape and use a footnote to provide more information. 3. Provide clear titles, subtitles and subscripts for a diagram. Possible uses: indicate position in whole set of diagrams, show importance, give reading clues, draw conclusion, and explain who/what/where/how/why/… 4. Annotations give answers to questions, focus the attention, and explain. Design labels and annotations so they stand out from the background but remain subordinate to the subject matter.

2.4.7 Graphics & Icons

Graphics & icons can be very useful to make diagrams appealing, especially for a non-technical audience. With graphics we mean freestyle artistic representations of objects or actions of modest size, for instance 1 square inch in print. Drawing packages for the PC usually come with libraries of this kind of graphics. With icons we mean the even smaller images of 32 by 32 pixels, which are today mostly known as recognition symbols for graphical user interfaces. Dreyfuss [7] and Modley [18] contain dated but still inspiring collections of icons.

The use of graphics and icons parallels the use of color. It's new. It has a strong visual impact. It is very dependent on cultural or company context. It's fuzzy. A difference is that it is a worthwhile option. You don’t have to use it. Another difference is that is it more difficult to come up with a new set of icons. Not many people have the ability to draw new icons or graphics. There are possibilities here for creative contributions in an architecture design project, comparable to creating new icons for a web site.

Table 2-9 Graphics & Icons Guidelines

Graphics & Icons
1. Use icons and graphics modestly. 2. Use icons: to speed search, for immediate recognition, for better recall, to save space, for graphic or spatial concepts, for visual appeal. 3. Don’t try to be funny. 4. Use graphics that are meaningful in the daily life of the viewers. 5. Adding icons to your boxes is useful for categorizing, like indicating all functions related to ‘finance’.

Figure 2-8 is an simplified example of a technical diagram in which icons are used to speed up recognition. The idea is that the same icons are used in the whole set of diagrams, and that an icon represents an abstract concept. Of course this only works if the icons are self-explaining to the viewer, or ‘easy’ explainable. In Figure 2-8 there are four different green objects, of which the middle two have something in common, as indicated by the color and the icons. The green objects have a number of unspecified relations to three yellow objects, of equal type, that are fulfilling three roles/functions/…

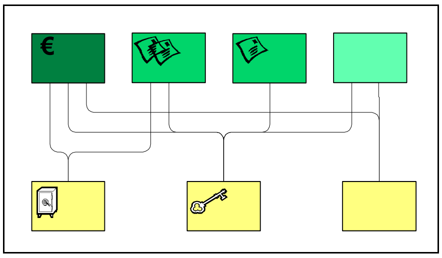


Figure 2-8 An example of a diagram with icons to speed up recognition.

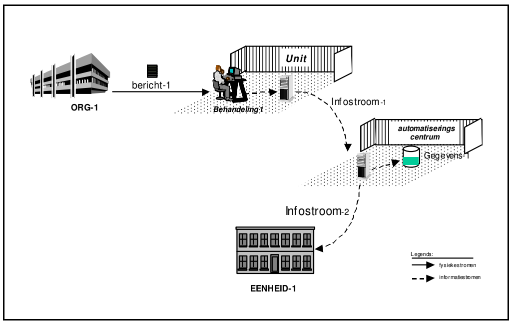


Figure 2-9 An example of a diagram with some freestyle graphics

Figure 2-9 is a simplified example of the use of free graphics. This is meant to be more appealing for non-technical viewers, but is less suitable for adding technical information. Of course this only works if the graphics are meaningful to the viewer. The details of the graphics should not distract from the meaning of the graphics.

2.5 Context and Design

Our focus in this paper is on the outward appearance of diagrams, as indicated in the various visual attributes. But in dealing with diagrams one soon comes to circumstantial factors, not specifically tied to one or more visual attributes. The ‘Context and design’ section of the guidelines tries to capture some of the circumstantial factors we encountered in our search for guidelines concerning visual attributes.

2.5.1 Design & Evaluation

The aim of this kind of guidelines is to support the architect who is planning or evaluating a set of diagrams from a general diagramming perspective. We do not take into account specific IT-architecture issues (types of views, certain concerns, stakeholders, etc). The essence is “making good diagrams is hard work”.

Table 2-10 Designing a set of diagrams Guidelines

Designing a set of diagrams
1. The use of diagrams should be designed, just as the architecture itself. It should grow during the design project, not added afterwards. 2. From the start of the design project, be on the outlook for good graphics which are meaningful to the stakeholders. 3. Create a clear structure in the set of diagrams. 4. The use of visual attributes should be designed with the whole set of diagrams in mind. 5. Don’t only prepare diagrams that show the good news. Create also diagrams which help the viewer to compare old and new, or to find possible blind spots in your report, to visualize for himself all your arguments… 6. Take into account the way of thinking of the organization. Adjust. Reach out. 7. Composing a diagram: start by identifying the most important information in the diagram. Allow no more than three to seven objects at this top level. Ideally, identify a single object to dominate the graphic. 8. Composing a diagram: formulate the specific questions a viewer could answer by looking at the diagram. 9. To evaluate a diagram the following list of general questions people ask while looking at diagrams can be useful: What is it? What is most important? How does it relate to other diagrams? How do I use it? Where am I in this diagram? Where does it start? What is the difference compared

to the current situation? 10. A series of consecutive diagrams can be used to: build up a complex diagram, simulate processes, progressively reveal more detailed information.

An example of a considered doubtful guideline in this category is "create graphic organizer as preview for coming structured information. Concept structure rather than report structure ➔ additional insight in / organization of subject matter".

2.5.2 Diagrams in the Architecture Report

To make the subject complete, here are some guidelines for incorporating diagrams in an architecture design report. In the text of this paper we have given an example of positioning diagrams and referencing meaningful to diagrams.

Table 2-11 Diagrams in the architecture report Guidelines

Diagrams in the architecture report
1. The distance in the report (number of pages) between a diagram and a reference to it should not be too big. A possibility is to repeat the diagram (this should be signaled in the caption). 2. Idea: repeat a thumbnail in the margin of the text. Highlight on the thumbnail the part of the diagram that is being treated in the text. 3. Always number your diagrams (figures). Text references to diagrams are always by number (not: above/below, at your left/right). References to diagrams are preferably placed at the end of a paragraph. See to it that all diagrams are referred to in the text. 4. Simply and directly inform the reader of diagram content and purpose. 5. Encourage the reader to look at the diagrams. Ask thought-provoking questions about them. Point to peculiarities.

2.6 Conclusion and Future Work

Looking at the various visual attributes is an effective means for finding ways to improve the readability of IT-architecture diagrams. Visual attributes we took into consideration are: visual hierarchical levels, layout aspects, coloring, forms and size, use of icons and graphics. Britton et al [4] have also expressed the importance of some of these classes of guidelines, in particular the use of a clear structure (hierarchy and layout) and motivating symbols (forms, icons and graphics).

A lightweight validation of our guidelines has been established in workshops with IT architects from various participating companies. This gave support for the practical usefulness of most of our guidelines.

Directions for further research include: maintaining and extending the guidelines (for instance the use of fonts), getting a better idea about the priorities of the various guidelines and about the way the visual attributes influence each other, differentiating different usages of diagrams, positioning the guidelines in a formal model of communication and perception, link the readability of diagrams to the readability of text. See Hargis [9] for developments concerning the readability of text.

Armed with an understanding of the most relevant visual attributes of IT-architecture diagrams we plan to start research into specific (new) types of diagrams that can be used to visualize specific aspects of an IT-architecture and into the way IT-architecture diagrams can be adjusted to better serve the interests of specific user groups (stakeholders).

2.7 Acknowledgements

This research is partly funded by grants from the Computer and Software Centre of the Dutch Tax and Customs Administration, ING Barings, Ordina Panfox, and Ordina Institute.

2.8 References

[1] J. Bertin, Semiology of graphics, The University of Wisconsin Press, 1983.

[2] Bernhard. H. Boar, Constructing Blueprints for Enterprise IT Architectures, Wiley, 1998

[3] Jan Borchers, Oliver Deussen, Arnold Klingert, Clemens Knörzer, Layout Rules for Graphical Web Documents, Computers & Graphics, Vol. 20, No. 3, 1996

[4] Britton et al, Evaluating the intelligibility of diagrammatic languages used in the specification of software, First International Conference, Diagrams2000, Edingburgh, 2000

[5] A. Brown et al (Eds), Describing Software Architecture with UML, Workshop at the 23rd International Conference on Software Engineering, Toronto, ACM, 2001.

[6] Hideaki Chijiiwa, Color Harmony, Rockport Publishers, 1987

[7] Henry Dreyfuss, Symbol Sourcebook: An Authoritative Guide to International Graphic Symbols, McGraw-Hill, 1972

[8] M. Fowler, UML Distilled: A Brief Guide to the Standard Object Modeling Language, Addison-Wesley, Reading, Massachusetts, 1999.

[9] Gretchen Hargis, Readability and computer documentation, ACM Journal of Computer Documentation, August 2000, Vol.24, No. 3.

[10] Guido van der Harst, Rob Maijers, Effectief GUI-ontwerp, Academic Service, 1999 (in Dutch)

[11] C. Hofmeister, R. Nord and D. Soni, Applied Software Architecture, Addison-Wesley, Reading, Massachusetts, 1999.

[12] W. Horton, Illustrating Computer Documentation, Wiley, 1991.

[13] IEEE, IEEE recommended practice for architecture description, IEEE Standard 1471, 2000.

[14] Henk Koning, Guidelines concerning readability of IT-architecture diagrams, to be found on http://people.cs.uu.nl/koningh/ 2001

[15] Henk Koning, Guidelines readability: example diagrams from internet, to be found on http://people.cs.uu.nl/koningh/ ,2001

[16] P.B. Kruchten, The 4+1 view model of architecture, IEEE Software 12 (6) (1995) 42-50.

[17] Jill. H. Larkin, Herbert A. Simon, Why a Diagram is (Sometimes) Worth Ten Thousand Words, Cognitive Science 11, 65-99 (1987)

[18] Rudolf Modley, William R. Myers, Diana G. Comer, Handbook of Pictorial Symbols: 3250 Examples from International Sources, Dover Pubns; 1977

[19] James Martin, Carme McClure, Diagramming Techniques for Analysts and Programmers, Prentice-Hall, 1985

[20] Narayanan, Diagrammatic communication: a taxonomic overview, Technical Report CSE97-06, December 2, 1997

[21] D. A. Norman, The Psychology of Every Day Things, Basic Books, 1988.

[22] Software Engineering Institute, Workshop on Software Architecture Representation, 16-17 January 2001, http://www.sei.cmu.edu/publications/documents/01.reports/01sr010.html

[23] Dennis Smith, Bill Thomas, Scott Tilley, Documentation for software engineers: what is needed to aid system understanding, workshop at SIGDOC 2001, Annual ACM Conference on Systems Documentation, http://doi.acm.org/10.1145/501516.501570

[24] D. Soni, R.L. Nord and C. Hofmeister, Software architecture in industrial applications, Proceedings 17th International Conference on Software Engineering, ACM, 1995, pp 196-207.

[25] Robert Spence, Information Visualization, ACM, Addison-Wesley, 2001

[26] H. van Vliet, Software Engineering: Principles and Practice, John Wiley & Sons, Chichester, 2000.

[27] Colin Ware, Information Visualization, Morgan Kaufmann Publishers, 2000

[28] Martijn van Welie, Task-based User Interface Design, Master Thesis, SIKS Dissertation Series No. 2001-6, electronically available at http://www.cs.vu.nl/~martijn/gta/docs/Welie-PhD-thesis.pdf

3 Real-life IT architecture design reports and their relation to IEEE Std 1471 stakeholders and concerns

Henk Koning Hans van Vliet

Faculty of Science

Vrije Universiteit, Amsterdam

De Boelelaan 1081a,

1081 HV Amsterdam, The Netherlands

henk@cs.vu.nl hans@cs.vu.nl

ABSTRACT

Architectural designs are an important means to manage the development and deployment of information technology (IT). Much debate has been going on about a proper definition of architecture in IT and about how to describe it. In 2000, the IEEE Std 1471 proposed a model of an architecture description and its context, which has been greeted with a warm welcome by many professionals in IT, but has not been applied much yet. In this paper the distance between IEEE Std 1471 and current practice is investigated. We have studied four real-life architecture descriptions from the practice of a bank and consultancy firm. These documents propose strategic decisions about application portfolios and were compiled without reference to IEEE Std 1471. Our research questions were: which parts of the document are, in the perception of the authors of those documents, relevant to which concerns of stakeholders? And, does this 'relevancy pattern' suggest an alternative organization in concern-related views? In other words, can the existing documents be (manually) re-engineered to IEEE Std 1471 documents? The answers to these questions enable authors to communicate more effectively to the stakeholders and can be input to future automated document generation.

We found that the 'relevancy pattern' is very scattered, and that an alternative organization is not evident. Most concerns are addressed by a relatively small, but each time very different, subset of the document. So re-engineering these documents to IEEE Std 1471 documents would incur an almost complete rewrite. Our research makes it very understandable that readers complain about too much information. Some stakeholders might well have difficulty finding the information of their interest.

The authors of the architecture documents found this investigation a worthwhile exercise, one which they think could be developed further into an evaluation instrument for this type of

The original publication is available at www.springerlink.com, see http://dx.doi.org/10.1007/s10515-006-7736-6.

documentation. Conversely, authors of architecture documents do well to make their stakeholders and their concerns explicit up front.

3.1 Introduction

Architecture is a relatively new branch within software engineering. IEEE Standard 1471 [10] defines it as "Architecture is 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". Common components of architectural descriptions are: levels or domains which divide the area of operation into manageable parts, various types of components (user interface components, security components, data management components, etc.), complex relations and behavior of all these components to deliver together the required functionality, ways to build/buy/maintain the components, planning phases (current, migration, target).

Van Vliet [19] places the architecture design phase in the software life cycle between the requirements engineering and other design phases. During the architecture design phase, the interests and concerns of all stakeholders are taken into account by the architect to come to a well-balanced solution. The result of the architecture design phase is a series of major design decisions that put constraints on the building process as well as the product delivered. This may apply to one software product, but it may also apply to several software products. The architecture conceived plays a pivotal role during the whole life time of a system. It guides system development and evolution. In particular, the architecture description plays a key role in a reengineering project, if such documentation is available.

Readers of architecture design documents are very diverse. They may be system designers, reengineers, programmers, users, managers of user departments, managers of IT departments, auditors, and fellow architects. Architecture design documents are artifacts that serve to communicate the ideas of the designer to the stakeholders. Current practice is that designers of architectures are problem-driven. After arriving at a balanced solution that solves the problem, they write down their solution in a structured way. This can be a structuring devised for a single project or a structure following a known framework such as those of Kruchten [12], Soni et. al [16], or Boar [3]. Traditionally, architecture documents are organized around topics like hardware, software, organizational issues, and other broadly recognized topics. This may be very good for stakeholders whose main concerns are either hardware, or software or organizational issues, but many concerns will crosscut the traditional organization. It is here that we seek to find better ways of organizing architecture descriptions taking the concerns of stakeholders as starting points and working within the framework of IEEE Std 1471.

The organization of this paper is as follows: in section 2 we lay out the research setting. In section 3 we give a description of the approach we took and the activities we performed. This section is an optional intermezzo and can be skipped over. In section 4 we present the relevancy patterns we found in the investigated documents. In section 5 we discuss these results and draw conclusions. We end in section 6 with a summary of the main conclusions and ideas for possible future work.

3.2 Research setting

In this section we lay out the research setting. We first introduce IEEE Std 1471 and develop our research questions from there. We then say a few words about the companies that were involved and the actual documents that were studied, and close with pointers to related work.

3.2.1 IEEE Std 1471

In this paragraph we give a short introduction of IEEE Std 1471, and we state our position towards this standard.

IEEE Standard 1471 describes a model of an architecture description (AD) and its context [10]. On page 1, it says: "The purpose of this recommended practice is to facilitate the expression and communication of architectures". On page 2, it says: "Furthermore, it establishes a conceptual framework of concepts and terms of reference within which future developments in system architectural technology can be deployed. This recommended practice codifies those elements on which there is consensus; specifically, the use of multiple views, reusable specifications for models within views, and the relation of architecture to system context."

For this study, the central 'terms of reference' in the IEEE 1471 conceptual model are 'views', 'viewpoints', 'stakeholders' and 'concerns'. An 'architectural description' consists of 'views' that are made according to a 'viewpoint' (see Figure 3-1). A viewpoint on the one hand prescribes the content and 'models' to be used, and, on the other hand, it indicates its intended 'stakeholders' and their 'concerns'. Viewpoints and views have a one to one relationship. Viewpoints can be saved as library viewpoints for future projects. A stakeholder can have more than one concern. Concerns can be relevant for more than one stakeholder. In the rationale the architect explains his design choices, why he has chosen these particular viewpoints and what has not been solved. A system fulfills one ore more missions and functions in a certain environment.

The standard lists a number of essential stakeholders and concerns, and gives examples of the use of architecture descriptions and of some viewpoints. The standard gives no general guidance in defining viewpoints. It only states that a viewpoint addresses a set of related concerns and that the viewpoints together should cover all the concerns of the stakeholders.

IEEE 1471 is often quoted within the architecture design community (see for instance TOGAF [18], Clements [4] p. 361, de With [21]) and the proposed terms of reference are widely used and have replaced possible other terms. For this we compliment the authors of the standard.

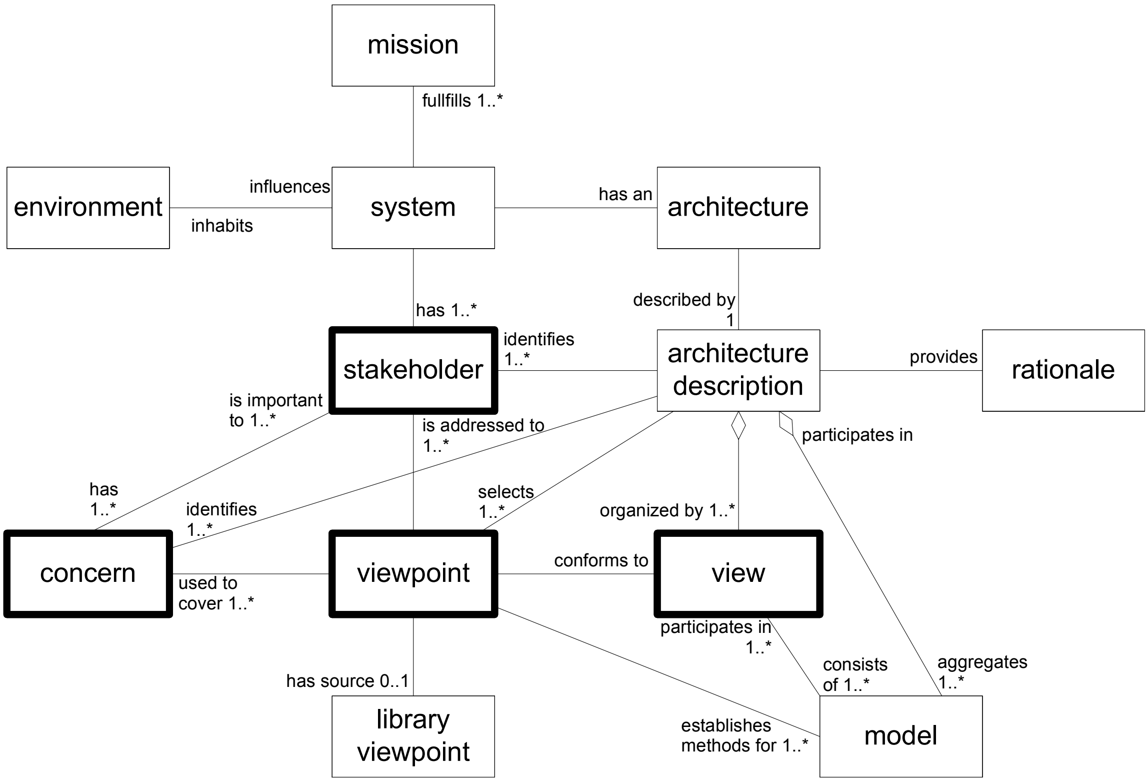


Figure 3-1 The conceptual model of the IEEE 1471 standard for architectural descriptions (essential concepts for this study are indicated in bold,)

Our position towards IEEE Std 1471 is as follows.

With respect to our interest in communication of architecture, the main contribution of IEEE 1471 is the explicit orientation on stakeholders and concerns. Following the path from his recognized concerns via the prescriptions in the viewpoint a stakeholder should be able to find the information of his interest in the views. For the architects IEEE Std 1471 is a stimulus to be very conscious of the concerns of the stakeholders and this helps him to shift away from a possibly too big emphasis on technical aspects.

Having said this, we still feel that the standard lacks vision on effective communication. For instance, being able to find all the information may in practice mean not finding the information. If a stakeholder has to refer to many different views in an architectural document and has to assemble a coherent picture on his concerns himself, this may in practice mean he will not go after this information and try to live with his best guesses. The same goes for information that is stated in terms or diagrammic presentations that the stakeholder is not familiar with. We feel the IEEE Std 1471 should be extended with guidance to achieving qualities like “accessibility” and “understandability”.

Another thing to point at is the lack of quantitive information in the standard. This applies to sample data from real-life architecture descriptions that follow the standard, as well as to expectations about the future architectural descriptions that will be compiled following this standard. Do the authors expect on average 5 viewpoints in an AD, or 25, or 100? How many concerns does a stakeholder have? How many viewpoints are necessary to address a concern? Not formulating at least an expectation leaves open too much room for interpretation of the standard.

3.2.2 Research questions

Since producing new architecture design documents following IEEE 1471, or re-producing existing ones, was beyond our possibilities, we decided to study architecture documents from the pre-IEEE 1471 era. Current practice is that architects produce one large document, from which the many stakeholders take what they need. See Figure 3-2 for an informal illustration.

Our interest in communication led us to hypothesize about the possibility of chunking architecture documents such that each

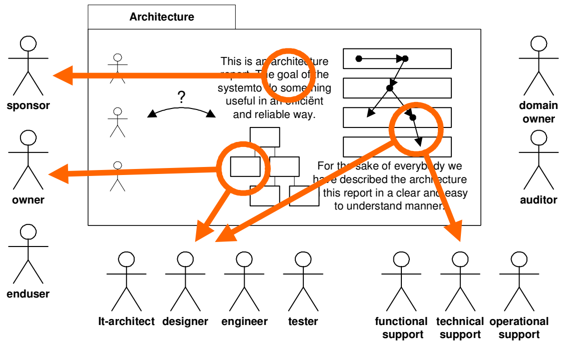


Figure 3-2 One large document, from which the many stakeholders take what they need

stakeholder is only confronted with information that addresses his concerns.

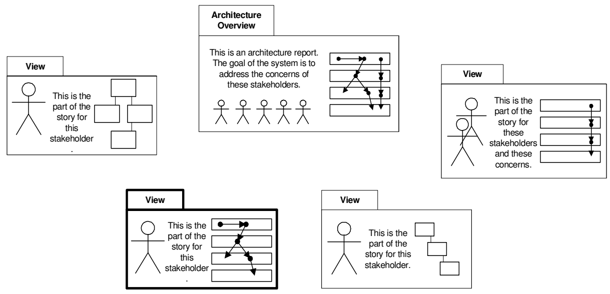


Figure 3-3 Breaking up the description according to stakeholder concerns

Figure 3-3 illustrates a situation in which a report containing all the information is split up in stakeholder oriented (concern oriented) chunks. The stakeholder is closely involved with this information, which is indicated by the small figures (one or more) standing in the report. A central, relatively small overview could still serve as a starting point for everybody.

The two main questions we address in this paper are:

1. What is the relevancy of the elements of the architecture documents for the perceived stakeholders and their concerns?

2. In the perception of the authors of the architecture documents, does the relevancy pattern warrant breaking up the description in stakeholder related chunks?

Now we will represent these questions in a more formal way. In

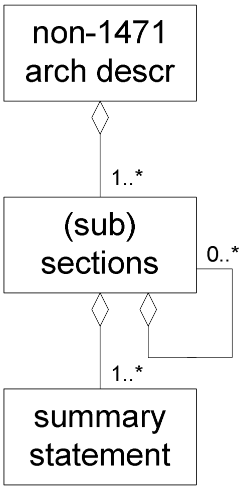


Figure 3-5 model of a non-IEEE-1471 document

Figure 3-5 we have modeled a non-1471 architecture description as a tree structure of sections and (sub)sections. For this research we have summarized each section of the investigated reports, more on this in section 3.3. In Figure 3-4 we relate the model of a non-1471 document (on the right) to essential concepts of the model of IEEE 1471 (on the left) by introducing two associations: one links the existing sections to re-engineered concerns of re-engineered stakeholders. The instances of this association form the answer to research question 1. Question 2 asks whether the relevancy pattern suggest new re-engineered viewpoints and views. If so, then second association links the old document organization to the new document organization in views.

3.2.3 The companies and the reports

The reports studied in this research activity were made available by two companies. One of them, CompA, is an international bank which attaches great importance to architecture to manage its very complex and diverse IT operations. Over 10,000 people are working in their IT departments worldwide, among them several hundred architects. The first two documents investigated were internal documents of this company. We will call them

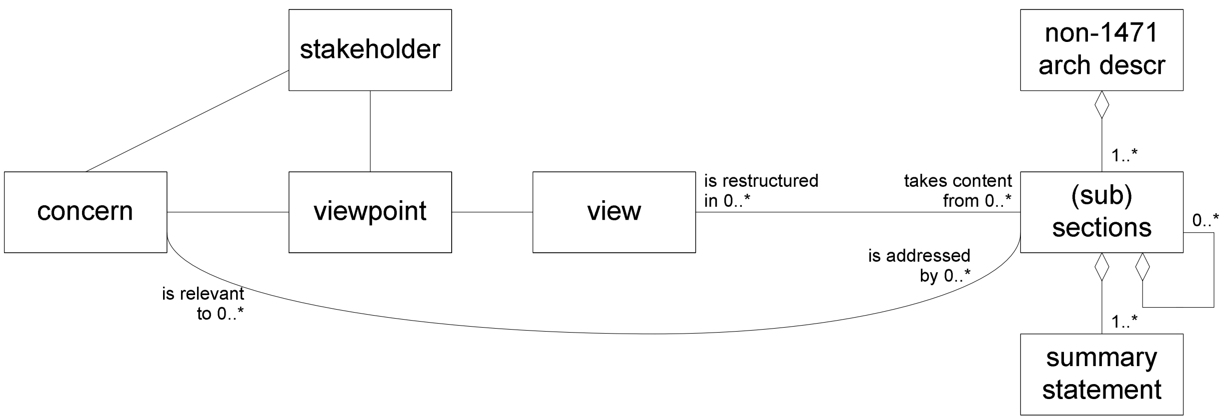


Figure 3-4 relating a non-1471 architecture description to essential concepts of IEEE Std 1471

DocA1 and DocA2. Two representatives of this bank participated in our research. One was involved in the realization of DocA1. The other was the co-author of DocA2.

DocA1 is a 90-page report, describing the efforts to enable an application-to-application cooperative processing for the complete European region, the largest region of the bank. The means to achieve this are, first, the realization of a highly reliable message bus which connects all sorts of platforms and, second, a transformation of all main applications into small pieces of shareable functions called services. This report is part of a larger plan to cut the yearly costs of computer operations by 300 million Euros.

DocA2 is a 150-page report, describing the effort to greatly reduce the number of applications and to centralize the computer processing environments for one of the bank's business units, which makes up 70% of the bank's activities worldwide. It gives engineers an architectural perspective on the 200+ applications currently in use and proposes a limited set of 15 new ones. And it extensively discusses the issues involved with network access to one global processing plant, new interfaces to other business units, links to local businesses and governments in various countries, and centralized global maintenance procedures. Currently, a team of over 150 people is permanently allocated to realizing these plans.

The second company, CompB, is a professional IT-consulting firm. It has been developing a vision on managing and documenting large IT processing environments for some years. All their consultants are trained in the methodology of the company. CompB made available documents from two of their customers, CompB1 and CompB2. There were also two representatives for CompB. One was the author of DocB1 and the other was the co-author of DocB2.

CompB1 is running a chain of travel agents. CompB1 has been very successful in the market, has grown steadily, but lacked an overview of all the software applications that had been developed over the years. DocB1 is a 135-page report, giving this recovered architectural overview. It lists issues to be solved and proposed solutions, a breakdown of the main business functions, a breakdown of the data, the allocation of business functions to departments, and the level of support the current applications give. The report ends with a proposal for new or changed applications for the coming years.

CompB2 is a government agency managing the money flows involved with several regulations in national law. Over the years, these regulations have been adopted by the government and serviced by this agency in diverse ways. The agency wants to start a new, more efficient way of working, with a modern and flexible software support (envisioned to support future regulations). DocB2 is a 62-page report laying the organizational foundation for the new software support. It offers an architectural analysis of the current situation, which consists of the categories of regulations which will be supported, the commonalities and differences between current regulations, and between current work processes. It then proceeds with a proposal for a new generic work process. The report ends with a chapter containing details of the new multipurpose work processes.

3.2.4 Related work

As far as we know only Smolander and Päivärinta [15] have done research regarding the use of IEEE 1471. They examined the reasons for making architecture descriptions in practice. Interviews with various stakeholders of architecture in three companies showed that beside the traditional use as a starting point for system design, architecture documents serve to communicate, to negotiate and to capture knowledge. According to the authors, these other uses should lead to new viewpoints and new tool support. One of the recommendations of Smolander and Päivärinta is to do further research into "how communicative these descriptions appear to be in the varying contexts of their use" (p.124). Our research is in line with their recommendation.

Hilliard [7] tries to strengthen the formal definition of architectural views by translating concepts from the domain of designing program modules, like 'validity checking', 'encapsulation', 'integration', to the domain of designing architectural views. From our emphasis on stakeholder oriented communication, we appreciate 'encapsulation' as a mechanism to deliver the right information to the right person.

Clements et al [4] offer an elaborate roundup of software architecture modeling techniques that have developed over the years. They put these in a fixed framework that distinguishes module views, component-and-connector (C&C) views, and allocation views. Module views document the principal units of implementation, C&C views document the runtime units and their interaction, and allocation views document relationships between the system's software and its environment. They add to that advanced modeling techniques, many good pieces of advice on documentation, and a list of suggested stakeholders and their information needs. In our study we offer a technique to analyse the stakeholder information in a given situation, which can give support in deciding on which parts of the fixed framework to use or adjust.

3.3 Wandering around, Settling down, staying at it

This section may be skipped over; without it the rest of the paper still presents a complete report of our results. At this point we insert an intermezzo - "for whom it may concern". In this intermezzo, we describe the search to find a workable research method. We not only describe the successful activities, but also the wandering around to find them. It took us some time to find a proper and balanced way of dealing with the questions raised. Actually it was a search process till the very end. More than once, our ability to adjust our thinking and change directions was stretched to the limit. We think it is also interesting to see what did **not** work. Also, this section may help to properly assess the value of our findings and understand the limits of their applicability, and it illustrates the collaboration between researchers and practitioners in defining the research question, that is an integral part of the *action research* approach we follow [1][2][17]. In this approach researchers and practitioners work together to find improvements in a real life situation, which is too complex to transfer to a clinical research environment.

The activities were performed over a period of nine months, with a capacity of two-person days per week, overhead included.

Many things go through one's mind when seeking a way to do new things. The steps we took can be roughly characterized by these one-liners:

- Agreeing on a *research question* with all the participants
- Deciding on the *data* to capture and trying various ways to record them
- Unsuccessfully trying to let the participants do it *themselves*
- Doing it *ourselves* and learning a lot
- *Finishing* it ourselves, but not achieving the goal
- *Successfully* letting the participants do it themselves.

We had been thinking a lot about formulating a *research question* in the realm of architecture that would be potentially beneficial to the participating companies, but also recognizable to the research community. The applicability of the IEEE 1471 standard had both qualifications. The companies knew of the standard, had a positive impression of it, but had not yet worked with it in practice. So we prepared a proposal based on IEEE Std 1471, and suggested right away to go after the most challenging stakeholder as far as communication of architecture is concerned, the end–user. Our assumption here is that architectural decisions can have implications for end-users and should be discussed with them. We studied literature from requirements engineering, Kotonya and Sommerville [11], and user interface design, van Welie [20], to find the essential attributes for stakeholders and concerns. The participants later on choose to use free format descriptions of stakeholders and concerns rather than a fixed format. The participants found we had taken big steps in deciding on a subject and needed more time to decide on the research question to address. We designed an evaluation form for them to do a quick sketch of stakeholders and concerns addressed in a document.

In the next round they were convinced this was a good idea, but asked us not to limit the investigation to just end-users. A list of perceived common stakeholders was compiled by all architects involved in this study, which we briefly mention here to give you some idea: sponsor/investor, owner, end-user, domain owner, architect, designer, engineer, tester, functional supporter, technical supporter, operational supporter. When the documents were chosen this list proved not useful. The stakeholders were not so common after all. In section 3.2.2 we mentioned already the research questions eventually agreed upon.

Deciding on the *data* to capture brought to light that the IEEE 1471 model, though at first sight easy to follow, is in fact quite complex. We started on building a prototype application for data capturing in MS Access, with which the participants could identify and categorize elements of an architectural description, define IEEE 1471 stakeholders and concerns and specify the relevancy of the elements for them. We very soon were spending a lot of time precisely positioning data input fields and text labels, and other complex programming issues doomed (such as capturing n:m relations). Also, the participants would probably not be very enthusiastic about spending hours filling in very detailed data-entry screens. With all the uncertainties about the right way to go, that seemed not to be a good investment of our time.

Since one of us was familiar with MS Visio (from drawing architecture diagrams as an IT-consultant) we tried another route, and designed an interactive, graphical representation with four columns: stakeholder, concern, architecture statement, component, see Figure 3-6 for an example representation hereof. The first two columns were intended to capture the essential stakeholders and their (shared) concerns. The last two columns were intended to express the essential content of an architecture. Our idea was that we could ask the architects to summarize their own design reports in columns three and four. It was fun to make and had appeal. We had a few tryouts on our own which went rather well, but left us with some doubt about the precise semantics of the fourth column, is the concept 'component' addressing the heart of an architecture design?

We then tried to let the participants express the relation of document elements to stakeholders and concerns *themselves* by means of this graphical representation. We arranged a collaborative session with representatives of one company in which they could 'think aloud' and we would edit the diagram. It went not very well. It was difficult for them to formulate the main concerns by heart, and at the same time formulate the architecture statements to address them. They found it difficult to express the essence of the architectures by means of items in the 3rd and 4th columns of our scheme.

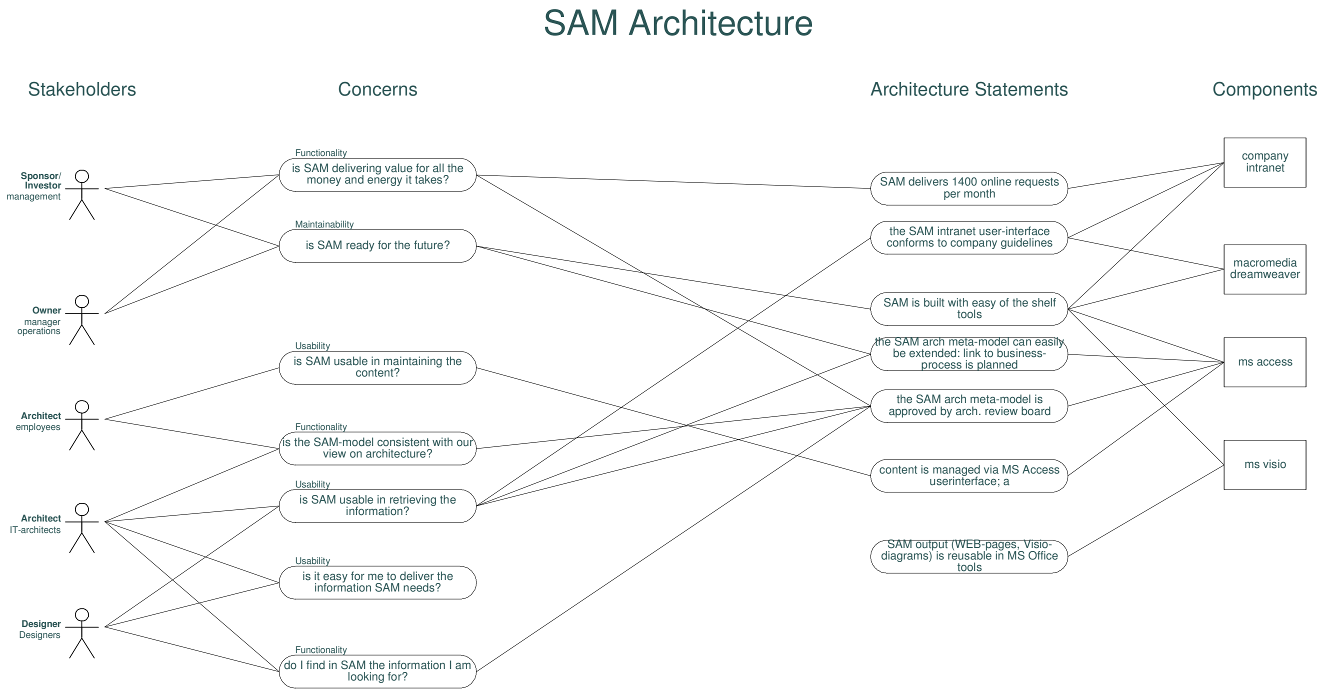


Figure 3-6 The experimental graphical presentation of stakeholder concerns related to architectural statements

So the only way forward seemed to be to first do an analysis all by ourselves. In the next session with the representatives of the other company, we asked them to give us a head start for our own analysis by objectively naming stakeholders and concerns per report (columns 1 and 2). This went rather well, but these head start data were eventually not used.

We then started to do an analysis of the document *ourselves* (which in the end proved to be very instructional for us). Again, how to go about it? After a lot of thought we started by simply making a résumé of one of the documents. This was straightforward. From our own experience in the field, the content of the document was very recognizable. After finishing the first content summary, we transformed it into a table having three columns: type of content, stakeholders and concerns and general remarks. The column 'type of content' was a vain attempt to arrive at a categorization of architectural statements, that could be linked to, for instance, types of graphical presentations (a leftover from our research into guidelines for readable diagrams in the year before). In the column 'stakeholders and concerns' our best guess was noted about what party in the company would be most interested in the content and why. Sometimes the report itself gave indications to this end. In 'general remarks' we put our questions concerning the content, our observations on the use of diagrams and our ideas about what makes an architecture design report readable. Summarizing all the four documents was a lot of work, tedious work.

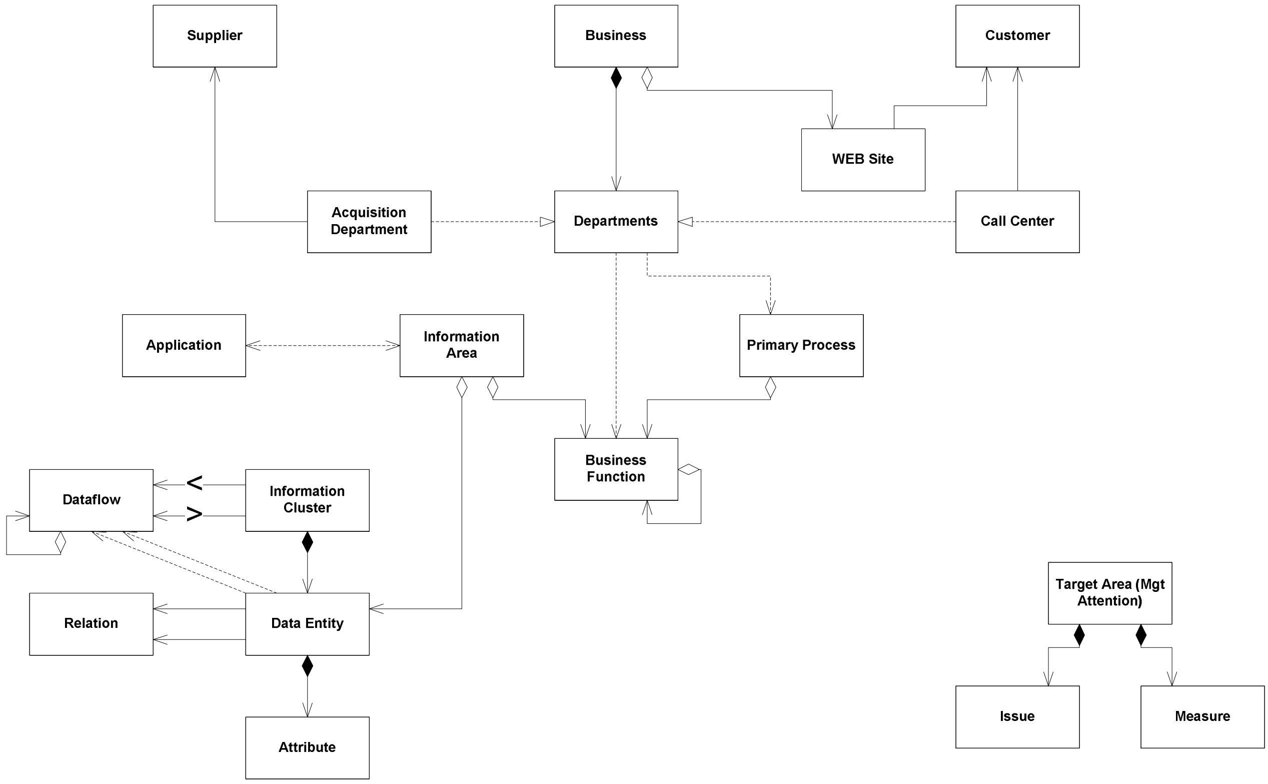


Figure 3-7 Map of the main concepts of an architectural design report in this study

At the next meeting with all the representatives we had analyzed in this way three reports fully, and one report half. At almost the last minute before the meeting we had prepared concept maps of each of them showing the main architectural concepts. Figure 3-7 is an example that shows the concepts that were treated in one of the design reports. It is a derivate of a more general conceptual model used by CompB and is similar in function to architecture meta-models as proposed by, for instance, Hofmeister et al [9]. The representatives did not have many comments on the analyses, but were pleasantly surprised with the ideas for making a report more readable and the concept maps. We were kindly asked to fulfill a consultant role in a project to devise seven new target architectures, and, to finish the analyses of the documents and draw our conclusions.

The feeling grew we were on the right track and valuable insight was coming out of this. We *finished* our part of the tables. At that moment, in our view, it remained unsatisfactory, mainly on two points. It proved very difficult to discern from the text of a report the main stakeholders and their interests. From our own experience, it was not difficult to discern content that would be of value for persons in the roles of manager, designer, programmer, etc. But were all these stakeholders present in the various companies? And what were exactly their responsibilities? So we had our best guesses, but no facts. And there was another thing. In our mind the question of what data to record had not been settled. Should we simply, superficially, record

Table 3-1 Relationship of chapters in DocA1 to the concerns of five stakeholders

	Stakeholders and Concerns									
	Business User Representative		CIO			Developer	IT-Architect		Project-leader	
Report chapters	do the applications fill my business needs?	what are the consequences of this architecture?	what projects are needed?	how can I make the IT cheaper?	what alternatives are available?	how does my project fit in the architecture?	how to advise for project approval?	how to help projects?	which components can be reused?	Total (max 9)
2 Overview		x		x	x	x	x	x	x	7
3 Overall Policies	x	x		x	x	x	x	x		7
4 Concepts and Blueprint		x		x	x	x	x	x		6
5 Defining Unified Services						x	x	x	x	4
6 Technical Interface Model						x		x		2
7 Usage Patterns						x		x		2
8 Non-functional Attributes						x		x		2
9 Security Model						x		x		2
10 API and Adapters Platform Strategy and Policies						x		x		2
11 Operational Management						x		x		2
12 Development Toolkits and Standards						x		x		2
13 Migration Considerations		x			x	x		x		4
Total (max 12)	1	4	0	3	4	12	4	12	2	

A checkmark behind a chapter in one of the columns means that the stakeholder needs the information from that chapter to know how his/her concern is addressed.

that some document element was relevant for a certain concern, as a mere yes or no? Or should we dig a bit deeper and try to figure out what this piece of information means to the concern? And what aspect of the concern does the information touch? We decided to keep it simple and only record a yes of no as the whole operation was already complicated enough.

The last step was to go back to the participants. This *successful* step is described in the next section, an introduction to the results of the analysis. At this point we want to say that this final step was easy going. The participants really got involved and even seemed to like the exercise. We had the feeling that it gave them a means to express something they maybe tacitly knew, but were not able to express so clearly before. The only thing not so pleasant for them was that in their own opinion evidently not all the stakeholders were interested in the whole report. Actually, almost no one was, but that already brings us to the discussion of the results.

3.4 The relation to stakeholders and concerns

This section shows, for the four architecture reports mentioned in section 3.2.3, the relationships between the various parts of the content, and the perceived stakeholders and their concerns. These tables represent the opinion of the persons who are co-authors of these reports or who are closely involved in carrying out the design decisions of the reports.

Table 3-2 Relationship of paragraphs in DocA2 to the concerns of five stakeholders

	Stakeholders and Concerns									
	Business User Representative		**CIO**			**Developer**	**IT-Architect**		**Project-leader**	
Report paragraphs	**do the applications fill my business needs?**	**what are the consequences of this architecture ?**	**what projects are needed ?**	**how can I make the IT cheaper?**	**what alternatives are available ?**	**how does my project fit in the architecture?**	**how to advise for project approval?**	**how to help projects ?**	**which components can be reused ?**	**Total (max 9)**
Introduction	x	x	x	x	x	x	x	x	x	9
2.1 Issues with current IT environment	x			x						2
2.2 Current Application Inventory	x			x	x					3
3.1 IT Architecture Objectives		x		x	x					3
3.2 IT Design principles		x		x			x	x		4
3.3 IFSA Compliance		x			x	x	x	x	x	6
4.1 Functional Breakdown	x							x		2
4.2 Data Model Overview	x							x		2
4.3 Target Application Portfolio	x	x	x	x	x					5
4.4 Major System Interfaces			x		x	x	x	x		5
4.5 Strategic versus Tactical Solutions		x	x	x	x	x	x	x		7
4.6 Special Facilities		x	x		x		x	x		5
5.1 Technical Standards				x	x	x	x	x	x	6
5.2 Technical Architecture Requirements				x	x		x	x		4
5.3 Technical Infrastructure		x		x	x		x	x		5
5.4 Technical Architecture of core banking application	x	x		x			x	x	x	6
5.4.1 Physical application and database						x				1
5.4.2 Client server model: this section						x				1
5.5 Technical Architecture of Other Applications	x	x		x			x	x	x	6
5.6 Interfacing architecture	x	x		x			x	x	x	6
5.7 Print/output architecture	x	x		x			x	x	x	6
6.1 IT Operations Processes	x	x		x	x				x	5
6.2 Operations Management Software	x	x		x	x				x	5
6.3 Security Architecture	x	x		x	x				x	5
7.1 Environments				x		x			x	3
7.2 Development Tools and Standards				x		x			x	3
7.3 Change control										0
8 Relation to other Service Centers		x			x				x	3
Total (max 28)	13	16	5	19	15	9	13	15	13	

The tables were compiled as follows: The participants were provided with tables containing summaries we had made of the reports. In separate collaborative sessions we asked the participants to mention the main stakeholders whose concerns had determined the content of the architecture. For each stakeholder we asked the participants to list the main concerns that would drive the interest in the report of that specific stakeholder. To keep things manageable, we asked them to try to identify not more than about eight concerns. We left it to the participants to decide on which level of content they wished to put checkmarks. The preference seemed to be for the chapter level and, in some cases, on the paragraph level. We had three separate sessions (DocA1 and DocA2 in one session, and DocB1 and DocB2 in separate sessions). All participants, except one with whom we lost contact because of a job change to another company, reviewed the outcome some weeks later and made minor changes.

Table 3-3 Relationship of paragraphs in DocB2 to the concerns of ten stakeholders

	Stakeholders and Concerns											
	Commissioner	Process architect	Process designer	Information Manager	Business domain architect	General Project Manager		Application Architect	IT domain architect	Functional Designers	IT-projectleader	
Report chapters and paragraphs	**how can I efficiently administer new regulations?**	**what does the solution look like?**	**can I produce a clear process design?**	**how do I prepare the organisation for using this solution?**	**what is the effect of this process architecture on my business domain?**	**does this fulfill the commision?**	**how to manage (pitfalls, risks, resources, scope)?**	**with which application (current, new) will we realise this solution?**	**what is the impact on the IT-support in my IT-domain?**	**can I produce a clear functional design?**	**what do I need for my statement of work?**	**Total (max 11)**
1 Introduction	x	o		x	x	x	x	x	x	x	x	10
2.2 Target vision	x	o			x				x			4
2.3 Current problems	x	o			x				x			4
2.4 Design goals		o			x			x	x			4
3.2 Considered regulations	x	o				x					x	4
3.3.1 Scope in business proces model		o			x		x		x	x		5
3.3.2 Relations to other projects		o			x	x	x	x	x	x	x	8
3.4.1 Products and services		o		x	x					x	x	5
3.4.2 Customer-to-customer processes		o	x	x	x			x	x	x		7
3.5 Demarcation relevant bussines functions		o			x				x	x		4
4.2 Current organisaton		o		x	x				x			4
4.3 Commonalities and differences in current workprocesses		o		x	x			x	x			5
4.4 Commonalities and differences in current use of information		o			x			x	x			4
4.5 Current IT support		o						x	x			3
4.6 Conclusions current situation	x	o		x	x	x	x	x	x			8
5.2 Target organisation		o		x	x				x	x		5
5.3 Target use of information		o		x	x			x	x	x		6
5.4 Target IT support		o		x				x	x	x		5
5.5 Conclusions target situation	x	o		x	x	x	x	x	x	x		9
6.2 Overview customer-to-customer processes		o	x	x	x			x	x	x		7
6.3 Process Send Form		o	x							x		3
6.4 Process maintain customer info		o	x							x		3
6.5 Process process forms		o	x							x		3
6.6 Process compile receipt		o	x							x		3
6.7 Process process claims		o	x							x		3
6.8 Process info request		o	x							x		3
Total (max 26)	6	26	8	11	17	5	5	12	18	17	4	

The results are given in Table 3-1, Table 3-2, Table 3-3 and Table 3-4. See section 3.2.3 for short descriptions of these reports. The second column of Table 3-3 represents concerns of the author himself.

We did not ask for details of the stakeholder descriptions and their roles. That information is not relevant for the pattern of the information retrieved from the documents. When specifically asked, all participants said that the information in the report was sufficient for the stakeholders to see how his concerns were addressed. The participants found producing the tables a worthwhile exercise, which can be easily repeated for other architecture documents to evaluate the document design or the final product. Almost all of the participants spontaneously remarked that if they had to produce documents per stakeholder with overlapping content, they would need an automated tool.

3.5 Observations

With respect to the tables, we observe the following:

- The pattern of the checkmarks is rather scattered, especially for DocA2 and DocB2.

Table 3-4 Relationship of chapters in DocB1 to the concerns of five stakeholders

	Stakeholders and Concerns							
	Information manager		CEO	User Management	Functional Coordinator		Developer	
Report chapters	is the IT support in good shape (within time and budget)?	how can I start / manage projects (outsourcing)?	can I keep and enlarge competitive advantage?	do we have the proper IT-support for our workprocesses (workable and fitting)?	how can I estimate the impact of changes (time & money on a functional level)?	how can I manage the developers with respect to application functionality?	what is the big picture?	Total (max 7)
1 What is architecture?				x				1
2 Target Areas	x		x	x				3
3 Business functions 'from aquisition to sales'	x			x	x	x	x	5
4 Conceptual data model 'from acquisition to sales'	x				x	x	x	4
5 Organization model and information areas	x			x	x		x	4
6 Application architecture (current and future)	x	x	x	x			x	5
Total (max 6)	5	1	2	5	3	2	4	

This seems to indicate that the interest in the report varies considerably. (We did some rough statistical measurements on the 'similarity' between columns and the result confirmed our observation.)

- Almost none of the stakeholders is interested in the full report. Some are only interested in very little of its content. This is contrary to the tacit assumption of many architects to be writing a report that is of general interest. Our conjecture is that if this study would be repeated at a more fine grained level, the 'percentage of interest' would even be lower.
- The very partial interest stakeholders have in the contents of the reports makes one wonder how easily they can find the information they need. In our opinion, this issue deserves further investigation.
- There is no correspondence between stakeholder interest and chapter divisions. This strengthens the idea that what is a logical structure for an architect is not necessarily a logical structure for a stakeholder.
- The intuitive approach taken to establish these tables makes us hesitant about doing extensive statistical analysis. We left the totals in the tables as a service to the readers.
- The architects liked the insight into the interest of stakeholders found in these tables. One architect came with the suggestion that this kind of table could be used as a 'reading guide' in the introduction of a report. Another architect remarked that an explicit attention to stakeholder concerns would have led him to produce less documentation.
- When there is a clear dividing line for splitting a large document, it can probably be noticed early in this way.

With respect to the main research questions (What is the relevancy of the elements of the architecture document for the perceived stakeholders and their concerns and does the relevancy pattern warrant breaking up the description in stakeholder related chunks?), the following observations can be made:

- We illustrate the relevancy patterns in the tables we compiled in a simplified way. This simplified way was necessary to prevent us from losing ourselves in complexity. It also seemed to match well the intuitive, broad-brush approach of the architects.
- The tables show that stakeholders are mostly interested in only part of the document, often less than half of it.
- DocA1 seems to have a clear dividing line for a breakup into one part of general interest and another part of interest to the developers. Such a clear division is not obvious in the other documents.
- For practical reasons our investigation was limited to the surface level of the current compartmentalization of the documents. The big differences in the meaning of the concerns of the various stakeholders make us believe that even stakeholders who refer to the same part of a document, actually look at different things.
- In DocB1, the chapters (3, 4 and 5), which describe the analysis of the current situation, seem to play a different role than the more decision-oriented chapters (2 and 6). The analysis chapters will mainly be used for subsequent application design and project control. The decision chapters are the basis for initiating the right projects.
- Distributing many copies of slightly different variants of a report would probably create a lot of confusion. There is also some organizational psychology involved: “Maybe I don’t need all the information, but at least I have the same information my colleagues have”.

With respect to IEEE 1471 we observe the following:

- IEEE 1471 leaves open the question of how to arrive at proper viewpoints. This openness is intentional, since architectural methods and organizations differ, and have their own rules for doing so. In our case, each column of the tables in section 4 represents a concern. Taking the columns of the tables in section 3.4 as the content of as many viewpoints will probably lead to documentation that is too fragmented and difficult to maintain.
- The chapters in the tables of content of the four reports show a clear grouping, mostly around some type of architectural concept, like ‘business functions’ or ‘applications’.

One could call this 'natural' organization "views" and declare each chapter to be a view. Compared to IEEE 1471 they still lack an explicit relation to stakeholders and concerns, but they do support the claim of IEEE 1471 that architectural documents are organized in views.

- In this small sample set of documents stakeholders and concerns are very different from one company to the next. This supports the choice of the IEEE 1471 design team to stick to a very open conceptual model.

Some general observations:

- This investigation very much underlines the value of being conscious of the relevancy of the architecture document for the different stakeholders and their concerns.
- Along the way we gained experience with representing the essence of an architecture report by means of concept maps, see Figure 3-7. We expect the use of concept maps to give visual support to the design of IEEE 1471 viewpoints by giving support in reasoning about which concepts are useful for addressing which concerns. Hilliard did something similar using UML [7].
- The variety of stakeholders, concerns, and content of the architecture document underlines the complexity of the practical situations in which architects work and shows that the 'art of architecting' is very situational.
- A possible future automated tool to generate stakeholder oriented documents from a central information base must be very flexible and adjustable to the needs of a particular project as they arise.

3.6 Conclusion and Future Work

Our main conclusions are:

- Many concerns are addressed by a surprisingly small part of the document. Concerns that refer to less than 25% of the document are no exception; see the tables of chapter 3.4.
- The pattern of stakeholder interest in the content of the documents is very scattered, which makes it difficult to devise a uniform scheme for breaking up the documents into smaller parts. Except for one document there is no clear dividing line for breaking up the documents.
- Specific stakeholders might well have difficulty finding the information of their interest (this is a documentation concern that definitely needs to be addressed).
- The authors of the architecture documents found this investigation a worthwhile exercise, an exercise which they think could be developed further into an evaluation instrument for this type of documentation.
- Conversely, authors of architecture documents do well to make their

stakeholders and their concerns explicit up front, and organize their documents accordingly, as opposed to merely following the solution structure.

We recommend experimenting with other ways of compiling architectural documents. For instance, at some point during the writing process, it might help to visualize its contents in a table whose rows represent the problem-solution ordering of the main architectural concepts and whose columns represent the stakeholders and their concerns. The entries of the table could then be filled with descriptive text which maps an architectural issue onto a concern of a stakeholder.

IEEE 1471 uses architectural views as the primary organizing principle for architecture documents. Furthermore, all relevant stakeholders and their concerns should be addressed somewhere, so this functions as a *completeness* principle. Our concern is the communication between architects and stakeholders. So we took the stakeholders and their concerns as a point of departure, and investigated how elements from several real-life architecture documents related to them.

Stakeholders and their concerns seem to be situational, and they have to be determined time and again. If these stakeholders and their concerns are not explicitly identified up front, chances are that the document is difficult to comprehend, for some or all of the stakeholders. We recommend using techniques as described in this paper, to do an early stakeholder and concern identification on proposed architecture descriptions. We are currently involved in workshops with IT architects to use this instrument in writing architecture documents.

Seen from an action research perspective, this research activity represents the diagnosis phase, which has given us important clues as to why stakeholders complain about architecture documents, and has given us ideas for improvement.

3.7 Acknowledgements

We thank the reviewers, Rich Hilliard, Johan F. Hoorn, Kari Smolander, Teck Yap and the anonymous reviewers of the ASE Journal, for all their very valuable comments.

This research is funded by grants from ING and Ordina Public Consulting.

3.8 References

[1] Avison, D., F. Lau, M. D. Myers and P. A. Nielsen (1999). "Action Research", Communications of the ACM, 42(1): 94-97.

[2] Baskerville, R. (1999). "Investigating Information Systems with Action Research", Communications of the Association for Information Systems, 2(19): 1-31.

[3] Bernhard. H. Boar, Constructing Blueprints for Enterprise IT Architectures, Wiley, 1998

[4] Clements, P., F. Bachman, L. Bass, D. Garlan, J. Ivers, R. Little, R. Nord, and J. Stafford, Documenting Software Architectures: Views and Beyond¸ Addison-Wesley, Boston, 2003

[5] M. Fowler, UML Distilled: A Brief Guide to the Standard Object Modeling Language, Addison-Wesley, Reading, Massachusetts, 1999.

[6] Gretchen Hargis, Readability and computer documentation, ACM Journal of Computer Documentation, August 2000, Vol.24, No. 3.

[7] R. Hilliard, Using the UML for architectural description, proceedings of <<UML>>'99 The Unified Modeling Language, Second International Conference, Lecture Notes in Computer Science volume 1723, Springer, 1999.

[8] R. Hilliard, Views as modules, position paper for the Fourth International Software Architecture Workshop (ISAW-4), 4 and 5 June 2000, Limerick, Ireland

[9] C. Hofmeister, R. Nord and D. Soni, Applied Software Architecture, Addison-Wesley, Reading, Massachusetts, 1999

[10] IEEE, IEEE recommended practice for architecture description, IEEE Standard 1471, 2000.

[11] G. Kotonya, I. Sommerville, Requirements Engineering, John Wiley & Sons, Chichester, 1998, ISBN 0 471 97208 8

[12] P.B. Kruchten, The 4+1 view model of architecture, IEEE Software 12 (6) (1995) 42-50.

[13] Software Engineering Institute, Workshop on Software Architecture Representation, 16-17 January 2001, Special Report CMU/SEI-2001-SR-010, http://www.sei.cmu.edu/publications/documents/01.reports/01sr010.html

[14] Dennis Smith, Bill Thomas, Scott Tilley, Documentation for software engineers: what is needed to aid system understanding, workshop at SIGDOC 2001, Annual ACM Conference on Systems Documentation, http://doi.acm.org/10.1145/501516.501570

[15] Smolander, K. and T. Päivärinta, Practical Rationale for Describing Software Architecture, Beyond Programming-in-The-Large, proceedings of 3rd Working IEEE/IFIP Conference on Software Architecture (WICSA3), August 25-30, 2002, Montreal, Québec, Canada, Jan Bosch et.al., ISBN 1-4020-7176-0, page 113-125.

[16] D. Soni, R.L. Nord and C. Hofmeister, Software architecture in industrial applications, Proceedings 17th International Conference on Software Engineering, ACM, 1995, pp 196-207.

[17] Susman, G. and R. Evered (1978). "An Assessment of the Scientific Merits of Action Research", Administrative Science Quarterly, 23(4): 582-603.

[18] TOGAF webpage Frequently Asked Questions, http://www.opengroup.org/public/arch/p1/togaf_faq.htm

[19] H. van Vliet, Software Engineering: Principles and Practice, John Wiley & Sons, Chichester, 2000.

[20] M. van Welie, Task-based User Interface Design, Ph.D. Thesis, Vrije Universiteit, Amsterdam, 2001, http://www.cs.vu.nl/~martijn

[21] Peter H.N. de With, Gert-Jan van Dijk, Architecture assessment for practical management of system architectures, paper to Landelijk Architectuur Congres, November 2002, Zeist, The Netherlands, http://www.serc.nl/lac/track6_software_en_embedded.htm.

4 A Method for Defining IEEE Std 1471 Viewpoints

Henk Koning Hans van Vliet
Faculty of Science
Vrije Universiteit, Amsterdam
De Boelelaan 1081a,
1081 HV Amsterdam, The Netherlands
Tel +31 6 2042 4278, Fax +31 20 598 7728
henk@cs.vu.nl, hans@cs.vu.nl

Abstract

With the growing impact of information technology the proper understanding of IT-architecture designs is becoming ever more important. Much debate has been going on about how to describe them. In 2000, the IEEE Standard 1471 proposed a model of an architecture description and its context.

In this paper we propose a lightweight method for modeling architectural information after (part of) the conceptual model of IEEE Std 1471 and defining IEEE Std 1471 viewpoints. The method gives support by outlining in textual form and in diagram form the relation of the concerns of the stakeholders to the architectural information. The definition of viewpoints can then be done with insight from these relations. The method has four steps: 1) creating stakeholder profiles, 2) summarizing internal design documentation, 3) relating the summary to the concerns of the stakeholders, and 4) defining viewpoints.

We have conducted a round of discussion and testing in practice in various settings. In this paper we present the feedback we received and propose improvements.

Keywords: architecture, IEEE Std 1471, viewpoints

4.1 Introduction

IT architecture is a relatively new branch within software engineering. IEEE Std 1471 (IEEE, 2000) defines it as "Architecture is 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". Van Vliet (2000) places the architecture definition phase in the software life cycle between the requirements engineering and design phases. In this phase the interests and concerns of all stakeholders are taken into account to come to a well-balanced solution.

Current practice is that designers of IT architectures are predominantly problem-driven. Design often is a fuzzy and non-rational process, see Parnas and Clements (1986), but after arriving at a balanced solution which solves the problem, the architect describes the solution in a structured way. This can be a one-time structuring or a structure following a known framework such as those of Kruchten (1995), Soni et al. (1995), or Boar (1998). Clements et al. (2003) offers many helpful models and guidelines for composing an architecture description. Using a one document framework for all stakeholders can mean for a certain stakeholder that the information that is relevant to his concerns can be very scattered, see Koning and van Vliet (2004).

In 2000, the IEEE Standard 1471 proposed a model of an architecture description and its context. It offers a high level generic model for architecture descriptions with explicit attention to the concerns of the stakeholders. In this

Journal of Systems and Software (www.elsevier.com/locate/jss), Volume 79, Issue 1, January 2006, Pages 120-131, http://dx.doi.org/10.1016/j.jss.2005.02.023

paper we offer support for the application of this model.

The organization of this paper is as follows: in section 2 we lay out the research setting. In section 3 we give a description of the method and show examples of the deliverables of each step. In section 4 we outline the validation activities and present the results for each step. In section 5 we draw our conclusions and propose improvements. Section 6 summarizes future work.

4.2 Research setting

In this section we lay out the research setting. We first introduce IEEE Std 1471 and state our position towards this standard. We then describe our project approach. We close this section with listing some assumptions on which our method is based.

4.2.1 IEEE Std 1471

IEEE Standard 1471 describes a model of an architecture description (AD) and its context (IEEE, 2000). On page 1, it says: "The purpose of this recommended practice is to facilitate the expression and communication of architectures". On page 2: "Furthermore, it establishes a conceptual framework of concepts and terms of reference within which future developments in system architectural technology can be deployed. This recommended practice codifies those elements on which there is consensus; specifically, the use of multiple views, reusable specifications for models within views, and the relation of architecture to system context."

Central 'terms of reference' in the IEEE 1471 conceptual model are 'views', 'viewpoints', 'stakeholders' and 'concerns'. An 'architectural description' consists of 'views' that are each made according to a 'viewpoint' (See Figure 4-1). According to the conceptual model a stakeholder is represented by his concerns.

A view is "A representation of a whole system from the perspective of a related set of concerns" (id, p.9), and a viewpoint is "A specification of the conventions for constructing and using a view. A pattern or template from which to develop individual

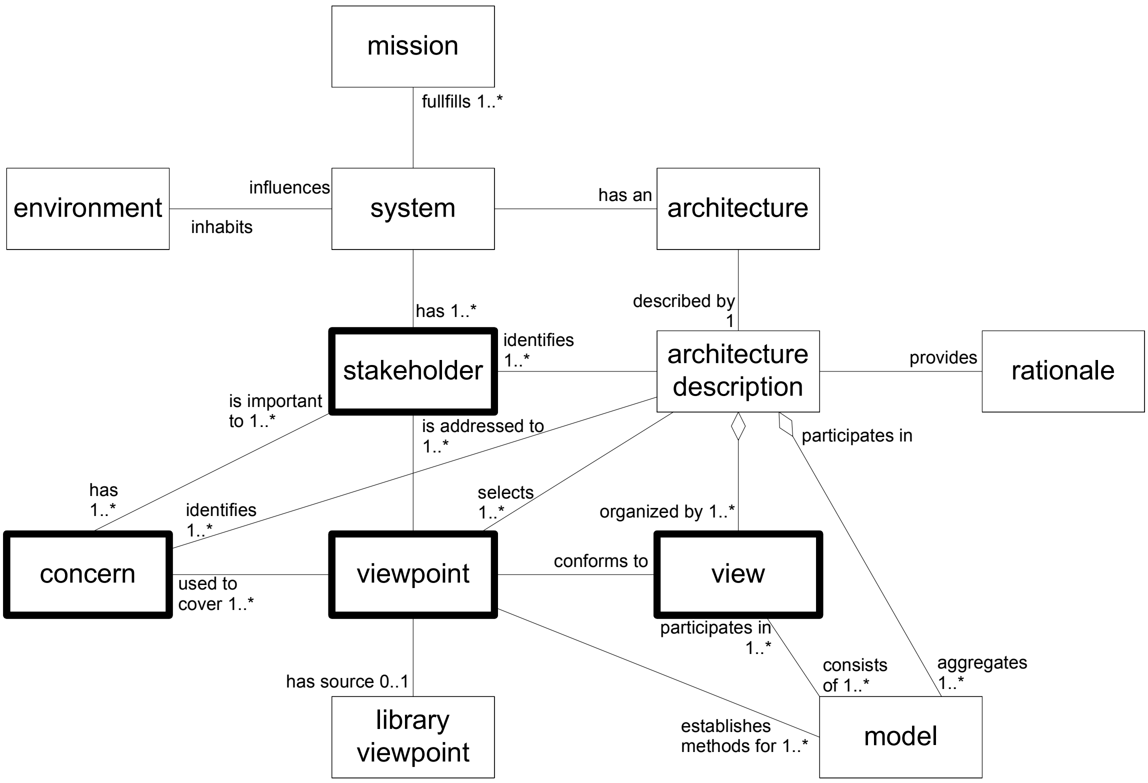


Figure 4-1 The conceptual model of the IEEE 1471 standard for architectural descriptions (essential concepts for this study are indicated in bold; where no cardinality is indicated it is '1')

views by establishing the purposes and audience for a view and the techniques for its creation and analysis." (id. p. 10). Viewpoints delineate the architectural information that is presented to the stakeholder. A viewpoint on the one hand prescribes the content and 'models' to be used, and, on the other hand, it indicates its intended 'stakeholders' and their 'concerns'.

The standard lists a number of essential stakeholders and concerns, and gives examples of the use of architecture description and of some viewpoints. The standard gives no general guidance for defining viewpoints. It only states that a viewpoint addresses a set of related concerns and that the viewpoints together should cover all the concerns of the stakeholders. There are no criteria given to decide on the 'relatedness' of concerns.

With respect to our interest in communication of architecture, the main contribution of IEEE 1471 is the explicit orientation on stakeholders and concerns. Following the path from his recognized concerns via the prescriptions in the viewpoint a stakeholder should be able to find the information of his interest in the views.

We also believe there are some drawbacks to this standard, see Koning and van Vliet (2004). We feel IEEE Std 1471 should be extended with guidance on how to achieve document qualities like "accessibility" to and "understandability" for the stakeholders.

Our research focuses on the definition of viewpoints as the leverage point to improve the quality of the architecture description, and more particular on improving the insight in the relation of the architecture design to the concerns of the stakeholders, before deciding on the viewpoints to use.

Application of this method contributes to meeting clauses 5.2 and 5.3 of IEEE 1471. It can also be used to construct library viewpoints or evaluate existing library viewpoints for possible use in a given situation.

4.2.2 Project approach

This research project follows an "action research" approach, see Baskerville (1999). In action research five steps are defined: diagnosing, action planning, action taking, evaluation, specifying learning. Our diagnosing of four real life IT-architecture documents has raised serious doubts about whether the stakeholders could find the information they needed. Our action planning resulted in the method for designing IEEE 1471 viewpoints described in this paper. Our action taking has been discussion and small scale testing.

The action research participants were IT-architects of two companies, ING and Ordina, and students from our faculty. ING is a Dutch international bank that attaches great importance to IT architecture to manage its very complex IT operations. Over 10,000 people are working in their IT departments worldwide, among them several hundred IT architects. Ordina is an IT-consulting firm. It has been developing a view on managing and documenting large IT processing environments for some years. Ordina Public Consulting has government and municipal organizations as customers. The students were taking part in a course on software architecture.

We were actively involved in the discussions and test sessions described in this paper as presenters and moderators. Notes were taken during all the sessions and shared with the persons present on the spot.

4.2.3 Assumptions

The method described in this document is based upon the following assumptions:

Designing an architecture is a fuzzy and non-rational process. For communication purposes the resulting design data need to be structured (Parnas and Clements, 1986).

Documenting the architecture is an activity that takes place in all stages of an architecture design project. For internal discussion or for intermediate discussion with stakeholders, parts of the problem statement and of the designed solution under consideration are described, altered, and described again. These pieces of description may have a varying degree of formality. There is a gradual shift from problem orientation to solution orientation.

Describing architecture at the moment requires a situational approach, which means, it is dependent on the peculiarities of the project at hand. Although many attempts have been made to standardize architecture descriptions, that is, to prescribe a fixed set of views, the current practice is that architects, for good reasons, make their own choices for each project.

Though the situational approach is common, it is not per se desirable. Where repeated use can be made of the same viewpoints, IEEE 1471 offers the possibility of storing and reusing viewpoints as library viewpoints.

An architecture description that is composed in a stakeholder oriented way is better readable for the stakeholder. Better readable means: the stakeholder can find more quickly the information that is relevant for him, and he can process that particular information more easily. Stakeholder orientation is determined by the structure of the document (the division in views and the outline within each view should be relevant to him), and the use of text and diagrams (words and graphics should be meaningful to him).

The smaller the number of views a stakeholder must consult to see how his concerns are addressed, the better it is. The smaller the amount of unnecessary information a view contains for a stakeholder, the better it is. Information that is only for internal use by the architecture team should not be communicated to the stakeholders.

Understanding and evaluating an IT-architecture by a stakeholder is basically a process of translating the IT-architecture concepts to his/her own concepts, and making inferences about possible situations or results that may occur by introducing the IT-architecture.

Diagrams play an important role in the communication of IT-architecture. Diagrams contribute mainly by giving an overview of components and relations and by giving support in making inferences, see Gyselinck and Tardieu (1998). Diagrams speed up the processing of the information and they aid in remembering the information.

Effective communication needs to be designed. The basic question is: what do you want to tell to whom?

Explicit representation (in text and graphics) of one's thoughts gives a better insight, leads to corrective thoughts and to a more complete design.

4.3 Viewpoint design method

4.3.1 Introduction

Our method for designing viewpoints consists of four activities that are, in principle, performed near the end of the design phase, before official decision taking and communication takes place. It embodies a roundup of what is already thought, said or written and puts it in a structure. If necessary, information can be added to get a more complete description. A partial application of this method earlier in the design project is feasible, and we actually expect that to happen in practice, but for testing purposes we don't want to make things too complicated at the moment.

Note: we like to mention here right away that the description 'near the end of the design phase' received some very straight criticism from the practitioners, even with the added nuances. Their comment: stakeholders and communication are essential throughout the whole design project. See further our 'conclusion' section.

An important aspect to consider is the level of detail with which these activities are performed. This can be restricted to what is necessary for properly clustering the architectural information in a stakeholder oriented way. The descriptive texts used in the method can be very terse, only to be understood by the designer himself. Our advice is to perform these activities with some speed in an intuitive manner, see what picture emerges and, if felt necessary, iterate to add details or improvements.

The activities to perform are:

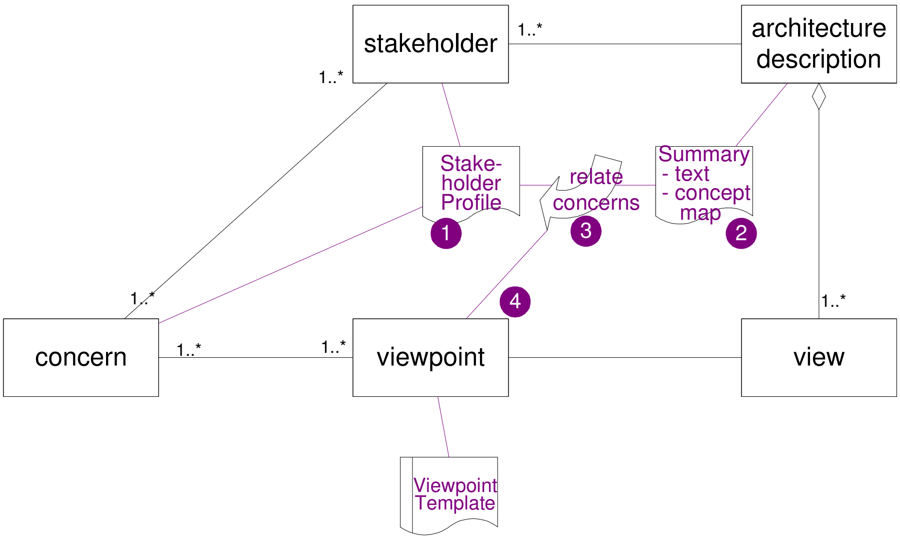


Figure 4-2 method viewpoints design, positioned to essential concepts of IEEE Std 1471

- Step 1 - Compile stakeholder profiles
- Step 2 - Summarize internal design documentation
- Step 3 - Relate summary of internal design documentation to concerns of stakeholders
- Step 4 - Define viewpoints

In Figure 4-2 these activities have been positioned in relation to the essential concepts of IEEE Std 1471. Steps 1 to 3 together create a ‘document design view’ of the architecture description to be made, which is then translated into IEEE Std 1471 viewpoints.

Basically, these activities provide a means to tinker for some period of time with the relation between the essential content of the architecture design and the concerns of the stakeholders. Various tools are provided to express and amend the thoughts of the architect, find omissions, seek words to express the perceived relation, etc.

We summarize in the next sections the method as it was presented after some time in the test period. The original description of the method, see Koning (2003), contains an extra step 5 ‘test result with stakeholders’, and has more details and more open advice and research questions. While presenting the method to the testers it was immediately clear that we had to restrict ourselves and we focused on the parts of the method that gave support for the clustering of architectural information in views.

The examples in the method description are from the ING target architecture project. The problem to be addressed there was that for a new financial reporting item, called ‘market risk’, the data is manipulated in many systems which leads to reporting errors and high maintenance costs. An extra viewpoint example is from a student assignment.

4.3.2 Step 1 - Compile stakeholder profiles

In this step a ‘stakeholder profile’ is compiled for each stakeholder that is relevant for this architecture design. A stakeholder profile is a simple table that holds descriptive text for five attributes: title, goals, tasks, concepts, concerns.

The goal of this activity is to make the stakeholder position explicit and to be able to reason about his information needs. The table expresses how the architect sees the stakeholder. It is a condensed résumé of the position of that stakeholder with respect to the problem domain, see Table 4-1. Uncovering new information is not the goal, but this activity may reveal that information is missing. If so, action can be undertaken to supply this information.

Table 4-1 Attributes of a stakeholder profile

Attri	Meaning

bute	
Title	Short recognizable description of role/function
Goals	The goal or goals of the role/function. A goal is a condition that must be reached or maintained.
Tasks	Logical grouping of all the activities that must be performed for the role/function.
Concepts	Objects that are relevant for the stakeholder and that make up his view of the world.
Concerns	Concrete interests or worries that guide the activities in the role/function and that determine which services are requested of other roles/functions.

The choice of these attributes was inspired by literature from requirements engineering (Kotonya and Sommerville 1998), and user interface design (van Welie, 2001). See Table 4-2 for an example of a stakeholder profile.

As far as the number of stakeholder profiles to produce is concerned, we consider five a good average and ten a lot.

Table 4-2 Example of a stakeholder profile

Attribute	Content
Title	CEO Finance
Goals	Deliver good financial planning and reporting
Tasks	- financial reporting - mgt reporting - product control (profit&loss) - raroc reporting
Concepts	Financial data, various kinds of reports, risks, budgets (medium term planning)
Concerns	- how do I deliver correct and reliable information? - how to receive/deliver financial data from/to other parties that are consistent with their reporting (Market Risks, Credit Risks)

For an even better understanding two profiles can be made for each stakeholder. One general profile expressing the position of the stakeholder in the organization, and one expressing the position towards the architecture.

IEEE 1471, clause 5.2 mentions some stakeholders and concerns that should be taken into account as a minimum.

The stakeholder profiles can be adjusted and further developed when used in the next steps.

4.3.3 Step 2 - Summarize internal design documentation

In this step the available, internal design documentation is summarized (if not done already). With internal design documentation we mean any recording of information that circulates within the design team and is part of the current design as far as it has progressed. This information can be structured according to some formal prescription or not, that is up to the designer.

The goal is to produce an overview of the architectural information that makes it possible to reason about the relation of this information to the concerns of the stakeholders. The overview should name parts of the information that can be allocated to views. It must be sufficiently clear to aid memory and be able to reason about it.

This step has two deliverables: a textual summary of the internal design documentation, and a map of the key architectural concepts.

The textual summary is a short bulleted list of the main architectural decisions. See for Table 4-3 for an example.

A number of five to ten statements seems reasonable.

Table 4-3 Example of textual summary of internal design documentation

Main architectural statements for new application architecture of Market Risks
• One system for all regions and business units • Use of a common data warehouse together with business unit Y • Application functions built as services that can be invoked over a message bus • Better systems • Great reduction in number of internal and external interfaces

It may seem strange to represent months of exploratory work and design deliberations in such a small list, but for communication it is necessary to create the ‘top of the pyramid’. While you are in the process of designing it is not always evident what currently your short list of statements is. Compiling this textual summary is a good way to become aware of what your current short list is.

This list can be adjusted and further developed when used in the next steps.

The textual summary will probably contain a number of terms that are key architectural concepts.

Architectural key concepts are the ‘things’ that come to surface when you ask ‘what kind of’ questions, like ‘what kind of components are part of the design?’, or ‘what kind of aspects did we look into?’. Examples of architectural concepts are: concerns, design principles, goals, applications, processes, products, infrastructure services, deadlines, money, design guidelines, etc. Concepts are part of the language of the architect. Concepts can be represented in a UML-class diagram. See Figure 4-3 for an example map of key architectural concepts (with added concerns of step 3). It is a mixed bag of topics that are for whatever reason relevant to the design.

The map represents a meta-model of the description of the architecture design. It strongly shows how elements of the design are related. It is a thinking tool, not a database design, therefore the presentation stays informal.

4.3.4 Step 3 - Relate summary of internal design documentation to concerns of stakeholders

In this step the results of step 1 and step 2 are related to each other. The goal of this activity is to make explicit the relation between the content of the architecture design and the concerns of the stakeholders. This relation is expressed in textual form and in graphical form. The texts reflect not only the fact that there is a relation, but also the essential reasoning that shows how the architectural statements address the concerns of the stakeholders.

The use in the next step (the design of the viewpoints, ‘who needs to know what?’) will determine the proper level of detail, which may not be easy to find right away. The text and graphics produced are of help in producing the final documentation and make that more smooth.

The textual expression of the relation of the content of the architecture design to the concerns of the stakeholders is expressed in a table. See Table 4-4 for an example. The first column contains the statements from the textual summary created in step 2. The table has two header rows: the second row contains the main concerns of the stakeholders. These concerns are derived from the stakeholder profiles created in step 1 by combining similar concerns. The first header row contains the stakeholders that share this concern.

At the crossing points of rows and columns, the question can be asked ‘is this architectural statement (row) relevant for this concern (column)?’ If so, that cell will be filled with some descriptive text that explains what the architectural statement means for this concern. It may also reveal some of the details behind the architecture statement that are relevant for this concern. In the empty cell above some descriptive text will be put that makes the (general) concern more specific for this architectural statement.

The table is a thinking tool. It is not necessary to add text to all the crossing points. To arrive at a balanced link between a statement and a concern some rephrasing may be needed of the statement or of the concern.

Making the architectural statements more specific is often more easy than making the concerns more specific. Making the concerns more specific is part of filling in the gap between the world of the stakeholder and the world of the architect, and it helps the architect in realizing what his statement means for the stakeholder.

Table 4-4 Example of content-2-concerns table (see text for description of the structure of this information)

Architectural statements	**Concerns**			
	CEO, CIO	CEO, Business Managers	Business Managers	CIO
	The current situation is very fragmented (many errors, high costs, long maintenance cycles), how can this be straightened out?	*How can new products be introduced more quickly?*	*How can we improve the quality of our work?*	*How can I reduce system development costs?*
	How do I simplify the IT operations?	*How can the information systems be changed more quickly?*		
One system for all regions and business units	One system will greatly reduce the complexity, but the migration can be very complex.	One system will bring time to market to 3 months (instead of 12)		
	How do I reduce system dependency?			
Use of a common data warehouse together with business unit Y	A common datawarehouse will reduce the number of interfaces from 50 to around 10.			
				How can I reuse available components?
Application functions built as services that can be invoked over a message bus				Approx 70 % of application functions can be reused by other systems over the message bus.
			How can the systems support the work more?	
Better systems			Better models will be developed that support more exceptions to standard operations	

The graphical expression of the relation of the content of the architecture design to the concerns of the stakeholders is based upon the concept map produced in step 2. This map is enriched with boxes that denote the stakeholders and their concerns. After that extra lines are drawn from the concerns to the architectural concepts in the map that are relevant to the concerns.

Figure 4-3 shows a Concept Map with the concerns of one stakeholder, related to the architectural concepts that are relevant for these concerns.

The concept map can soon become very messy when the concerns are added and the lines are drawn. If the tooling permits, this can be relieved by putting concerns in layers that can be made invisible. Descriptive text can be added to the lines, this gives the diagram a more immediate meaning.

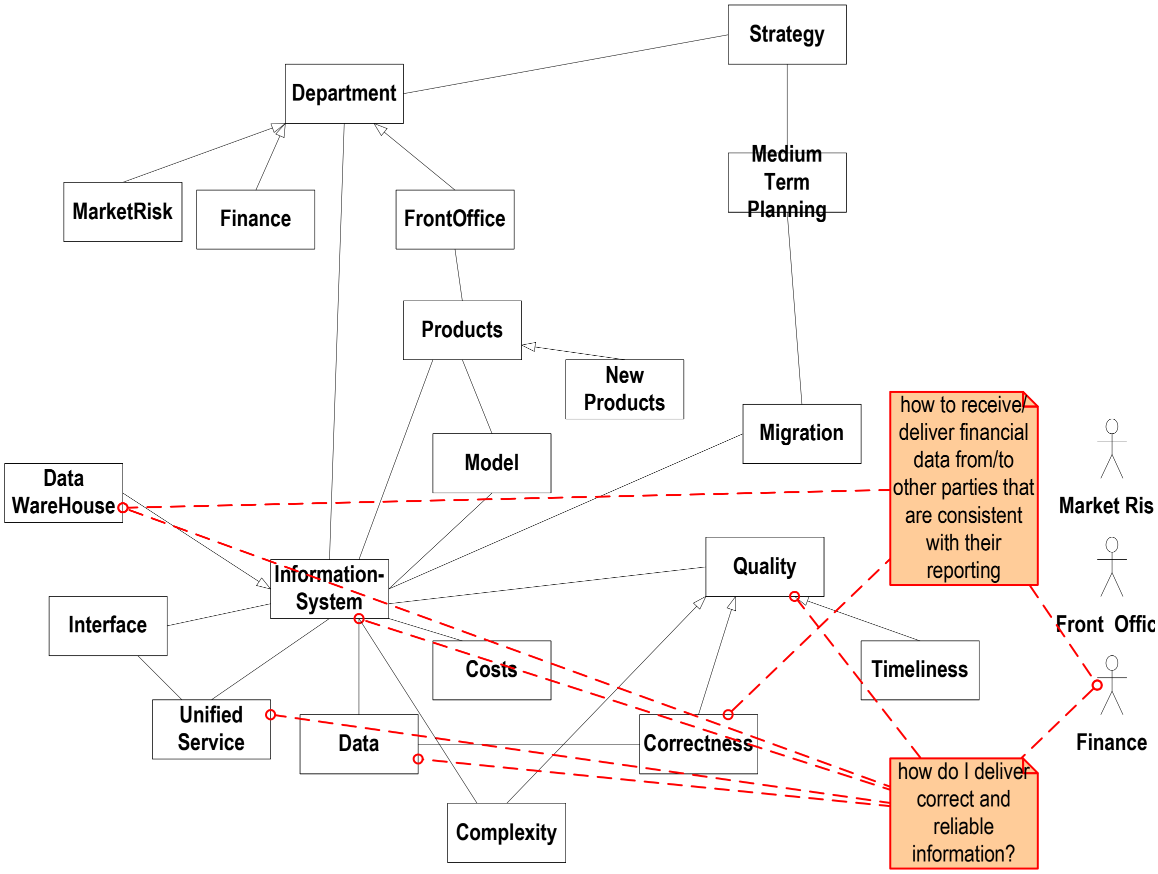


Figure 4-3 Example of a concept map with concerns

4.3.5 Step 4 - Define viewpoints

The goal of this activity is to define IEEE 1471 viewpoints in a way that fully takes into account what the architecture design means to (the concerns of the) stakeholders. The outline of text, the use of terms and graphics is geared to the stakeholders.

The primary decision to be taken is how information shall be clustered in views. Next a viewpoint is defined for each view to be produced. It seems only natural to capture the reasons for the choices made in this step as part of the rationale.

During our testing activities in the past months we offered two templates to the testers: a short one and an extended one, see Table 4-5 (the extended attributes are in italics). These templates were produced by us during the testing in reaction to the demands of the testers. To compile the extended list of viewpoint attributes we made use of IEEE 1471 Clause 5.3, Clements et al. (2003) (p 317) and Hilliard (2001). We left the tester freedom to decide which attributes of the template they wanted to use.

In the ING test case the IT-architects chose for only the outline to delineate the view to be made, see Table 4-6. Together with the content-2-concerns table this viewpoint determines the view.

A goal of designing viewpoints is to present to each stakeholder exactly the information he needs regarding his concerns. When he is presented with more information, it should be easy for a stakeholder to select the information in which he is interested. This information should not be scattered throughout the documentation. On the architect side there is the limited time available and the practicalities of the distribution of the final documentation.

For the extended attribute *Stakeholder oriented terms to be used* inspiration can be found in the list of concepts in the stakeholder profile. In the structure of the extended attribute *Outline of view* attention can be given to which concerns seem most pressing. Preferably the headings of the outline should be understandable for the stakeholder and serve by themselves as a summary of the architecture.

Table 4-5 Experimental Viewpoint Template with extended attributes

Attribute	Meaning
Title	Short recognizable description of the view
Stakeholders	List of stakeholders for whom the view is intended
Concerns	List of concerns that are addressed
Type of information	Free format description of the information that the view will contain
Presentation	The way in which the information will be presented, for instance as a book, as a report, or a slide-presentation, a help file, a website (with search engine?), etc.
Architecture concepts	*List of architecture concepts (from main concept map) to which the information is related. Used / needed attributes per concept.*
Formal Languages	*Which formal or informal language(s) will be used to describe the architecture?*
Modeling techniques	*Which formal or informal modeling technique(s) will be used to represent part of the architectural information?*
Analysis techniques	*Which analytical methods will be used to collect information that is needed?*
Outline of text	*Outline of the textual content of the view.*
Stakeholder oriented terms to be used	*List of terms from the world of the intended stakeholders that will be used in the descriptions in this view*
List of diagrams	*List of diagrams that will be used*
Stakeholder oriented graphics to be used	*List of graphical images (icons) from the world of the intended stakeholders that will be used in the diagrams*

Table 4-6 Example of viewpoint, using only the 'outline' attribute

Target architecture Market Risks – Finance View
Problem statement (based on concerns of CIO and CEO Finance)
How does the new proposed Market Risk Architecture relate to existing Finance Architecture?
Textual summary of proposed Market Risk Architecture
Which problems will be solved by the new architecture?
Concern 1 (one or more related columns from the content-2-concerns table for CIO or CEO Finance)
Concern 2 (idem)
Etc.
Which problems will be created by the new architecture?
Dependency of Market Risk from Finance

Table 4-7 shows an example of a (short) viewpoint, taken from one of the student cases.

Different modeling techniques can be combined in one view to represent all the information needed to address the concerns. See Clements et al. (2003) for a thorough description of proven modeling techniques. See Koning et al. (2002) for many practical guidelines concerning the design of diagrams.

Table 4-7 Example (short) viewpoint based on the template

Attribute	Content
Title	GUI view
Stakeholders	Judge, Registrar, Spokesman of police and justice, Representative of the lawyers
Concerns	- How to request the documents needed? - Registrar: How to update database?
Type of information	Visual impression of system. Possibilities of several (linked) screens.
Presentation	Drawings on paper
Analysis techniques	Execute scenario(s) on drawing(s)
Stakeholder oriented terms	Search, update, screen, button, task, document, functionality
Stakeholder oriented graphics	Screen, buttons, tables, search fields, icons

4.4 *Validation of the method*

4.4.1 Overview

Over a period of eight months our method was discussed and tried out in various settings:

At the university one student did an individual assignment on software architecture in which he redesigned an existing IT-architecture document. The method was also part of a course on Software Architecture in which groups of students worked as an architecture team to design the architecture of a new system.

We asked for comments in the public mailing list 'IEEE 1471 group interested' and we also received some other comments from individuals. One of us was involved in teaching courses on software architecture to software engineers of a Dutch company. The stakeholder profiles and the condensed viewpoint template were used in the assignments.

With Ordina Public Consulting (OPC) the method was presented to and discussed by a group of 5 IT-architects, two workshops were held in which the method was tried on generic stakeholder profiles and a generic architecture model (Jonkers et al. 2003), and a small group of consultants of Ordina has worked on redesigning an existing IT-architecture document.

The method was presented to two teams of two IT-architects of ING who were working on a target architecture. It was decided that with our assistance the team which was closest to producing a final report would apply the method to their case. The other team would take part in the sessions.

Because of the experimental nature of the method, we chose not to try it out in projects with a high commercial risk. That does not mean that the test situations were not serious and that the participants were not dependent on the outcome. We encountered a serious attitude in all the test situations.

In the sessions we presented the method and answered questions, but we did not take part in the actual work of applying the method. Most of the work was done by the participants outside the meetings, without us being present. In the meeting we did not primarily pay attention to the deliverables, but focused on the attitudes of the participating architects and students. Were the steps clear to them? Were they motivated to perform the next step? Did they find the process meaningful?

We gathered 25 pages of notes from the test sessions.

4.4.2 Findings step 1 - Stakeholder profiles

Step 1, compile stakeholder profiles, was generally received well by the participants in the sessions. They found it not difficult to do and it seemed to them an obvious step to begin with. We saw many good, clear profiles being made. A remark made was that this step should take into account IEEE Std 1471 clause 5.2 about stakeholders and concerns that should be identified at the minimum.

The main perceived immediate benefits were:

- It was helpful in imagining the stakeholders
- It was a good way to concentrate on each stakeholder and get to his/her essentials
- It made clear relevant differences between stakeholders
- Within the design team it was helpful in coming to a shared perception of the role of a stakeholder
- It is a fast way to present or check the perception with persons outside the team

Some comments on the attributes of the template:

- Many participants preferred to formulate the goal of a stakeholder as a main 'task', instead of as a condition to be reached or maintained
- Goals and tasks were helpful in finding concerns
- Some doubted the usefulness of the 'concepts' attribute, but others found it helpful in picturing the stakeholder.

- Some participants added an extra attribute, a short free format textual description of the stakeholder.
- The concerns we saw sometimes were a rephrasing of the tasks, 'how can I …', instead of real concerns.

4.4.3 Findings step 2 - Summarize documentation

Step 2, summarizing the internal design documentation, was more of a challenge to the participants than step 1, the stakeholder profiles. It was more of an effort, but the result was rewarding. In the findings there is marked difference between the textual part and the graphical part.

First the textual part. For the textual summary we observed that it required an effort, and that there were doubts about it being possible, but after it was done the participants liked their own summary and it served well in step 3. We saw many, in our view, good summaries, that outlined in a few sentences the essentials of a new situation.

With the student groups we saw some statements in the textual summaries that were a rephrasing of a concern, 'the system will be user-friendly', instead of indications of the IT-solution to provide this. We think this is partly because the summaries were made too early in the design process (requirements phase).

The main perceived immediate benefits of the textual summary were:

- It helped to articulate the essentials ('50 pages of information were reduced to three statements!')
- It forces to stay away from too much detailing
- It is a good way to express design decisions and come to a common understanding within the team

Then the graphical part. We pretty soon in some of the test sessions got negative feedback on the graphical part, the map of the main architectural concepts. It was considered difficult to produce and not useful. The relation to the textual summary was unclear.

In one case the concept map was perceived positive and in another there were concept maps already available. In these cases the perceived benefits were:

- It forces one to an even higher abstraction level than the textual summary
- It gives a good overview of many aspects

4.4.4 Findings step 3 - Relate summary to concerns of stakeholders

Step 3 was less of a challenge than step 2. It required some explanation, but once that was given, it was more a matter of work, interesting work. We again make a difference between the textual part and the graphical part.

Filling in the content-2-concerns table of the textual part seemed to be the most meaningful activity for the participants. It stimulated rethinking the architecture design and aroused a higher level of involvement. Producing a more specific wording of a concern with regard to an architectural statement was found more difficult than adding details to an architectural statement with regard to a concern. Also the limitation of putting it all on one screen was felt by some.

The main perceived immediate benefits of relating the textual summary to the concerns of the stakeholders were:

- It gives a new insight in what the architecture design means to the stakeholders
- It helps to stay focused on stakeholder concerns (instead of delving into technical details)
- It leads to strong, concise pieces of descriptive text
- It is a good way to express further details of the design decisions and come to a common understanding
- It is inspiring

As said with the findings of step 2, the concept map of the graphical part gave some problems. Where it was made or already available and the concepts were related to the concerns of the stakeholders, two problems were reported: the diagrams get messy (this was amended by working in layers) and after a while you don't know anymore why you have connected a concern to a concept (this was amended later by adding text to the connecting lines). On the positive side, it was reported that the diagram easily showed the differences in the information needs of the stakeholders and that they lead to strong, concise pieces of descriptive text.

4.4.5 Findings – step 4 - Define viewpoints

Step 4, define viewpoints, was probably the step least appreciated. For this step we cannot offer a list of main perceived immediate benefits.

There was some disappointment about the fact that this step required more than expected analyzing / thinking / designing. Another point seemed to be the fundamental question of 'why make viewpoints? Why not write the view right away?' Quite a few testers mixed view-description and view-content in the viewpoints. The terms 'view' and 'viewpoint' were found confusing by some. Where the views were produced, extra information was sometimes added without adjusting the viewpoints. The viewpoint template as described in 4.3.5 offered help, but also left many choices open.

Having said this, we can say we saw many viewpoints that, from first impression, outlined in a clear way the views to be made, which would address clearly stated concerns and which would be meaningful to the stakeholders.

We observed three analysis techniques used by the testers to make the transition from step 3 to step 4. Comparing the columns in the content-2-concerns table and combining similar columns in one view is one technique. Another approach is comparing the concerns that were used in step 3 to each other and creating groups of related concerns on gut feeling (but that does not make much use of the work of step 3). In one case we experimented with comparing the relations of concerns to concepts in the concept map.

One tester used the specifications of (sub)concerns in the Content-2-concerns table as headings in outlines of the views.

4.4.6 Findings concerning the whole method

From the test sessions we can report some miscellaneous feedback and observations that apply to the method as a whole.

We have received no reports on the method being used spontaneously in practice by the testers. One tester voiced his concern that application of the method in real life was too much work. Several participants in the sessions said that at the beginning they had some doubt about whether this method would really lead to results.

On various occasions the remark was made that communication is something that takes place during the whole design project and not only at the end. One participant suggested that the method should be applied several times during a design project on progressive versions of the design to ask comments from the stakeholders.

Because the method was presented early in the course the students applied it to requirements gathering and to the actual design work and not only for documenting results. A disadvantage of this is that the method is geared to converge to a small set of statements and which contradicts the investigative nature of the requirements phase. An advantage is that the students had a structuring mechanism that kept them on track.

On the whole the method forms a very straightforward process. During the process the already produced deliverables become a reference source. The steps in the method are clear and give structure to the working sessions. The attention in the sessions is held until the end.

At the detail level the method needs more explanation. The attributes of the templates and the meaning of the cells in the table need to be described better.

We often had to urge for speed and for an intuitive, iterative approach. After the first learning stage, the time needed to apply this method diminishes greatly. The thinking process provoked by the method seemed to be of particular value to the architects.

4.5 Conclusions

4.5.1 Method 'viewpoint design' as a whole

From the findings in the previous section we conclude that the proposed method 'viewpoints design' is meaningful to practicing IT-architects and leads to valuable results, but that improvements are necessary.

Application of the method is beneficial for expressing the positions of the various stakeholders, staying focused on the concerns of the stakeholders, reaching a higher abstraction level in the architectural design and for making clear what design information is needed to address which concerns.

Application of the method not only entails a roundup of existing design information, but brings with it some creative content production (more than we thought up front).

The techniques in the method can be applied not only at the end of the design project, but also during the project wherever design results need to be rationalized and communicated. We feel they are not suited to capture requirements, but they could very well be useful in design activities, wherever it comes to evaluating design alternatives against concerns of stakeholders.

The felt need to think about the way to communicate an architecture design is greater when: the architect has less experience, the architect is less familiar with the type of problem, there is no prescribed architecture framework.

4.5.2 Steps

The stakeholder profiles are not difficult to make and they function well to outline in a few words the position of a stakeholder.

The design summary is challenging, but gives a good grip on the design.

The content-2-concerns table of step 3 is a lot of work, but it is not difficult to make, it is a useful thinking tool and it is inspiring.

The concept map in steps 2 and 3 apparently is not suited for the average IT-architect busy in a design project, but it is probably more appropriate for people who have time and talent to reflect on architecture descriptions.

The viewpoint template works well to record the description of a view, but more support is needed for the transition from step 3 to step 4.

4.5.3 Improvements

The improvements most needed are: a better description of details of the method and more guidance in defining viewpoints. More guidance in defining viewpoints can consist of: guidelines for the transition from step 3 to step 4, example (library?) viewpoints, a more precise template, guidelines from other sources.

The concept map as the graphical design summary will not be made optional. We expect the concept map to give part of the needed extra support for the transition from step 3 to step 4.

4.5.4 IEEE 1471

Our findings indicate that the IEEE 1471 concepts 'stakeholder', 'concern' and 'view' are recognized and accepted by practicing IT-architects. The IEEE 1471 concept 'viewpoint' does not easily get operational significance. From the feedback received so far, we conclude that the average practicing IT-architect in ongoing projects prefers to work from IEEE 1471 library viewpoints, instead of creating viewpoints from scratch on his own.

From the unexpected remaining gap between step 3 and step 4 we conclude that the orientation on stakeholders and their concerns, as prescribed by the IEEE Std 1471 conceptual model, is in itself insufficient to delineate architectural views. More factors have to be taken into account, like, for instance, the inherent relations between the key architectural concepts as expressed in the concept map. We expect that an in depth analysis of dimensions in existing architectural frameworks will reveal more criteria for structuring architectural information.

4.6 Future work

Primarily we intend to improve the method, see 4.5.3.

The improved method will be tried out in real life architecture projects.

It is our desire to perform an in depth analysis of dimensions in existing architectural frameworks, see 4.5.4.

In the background we would like to collect more practical experiences with applying IEEE Std 1471 to deepen our understanding of the standard.

4.7 Acknowledgements

We thank Rich Hilliard and the other reviewers for their valuable comments. We thank the participating companies for their support.

4.8 References

Baskerville, R., 1999. Investigating Information Systems with Action Research, Communications of the Association for Information Systems, 2(19): 1- 31.

Boar, B.H., 1998. Constructing Blueprints for Enterprise IT Architectures, Wiley.

Clements, P., Bachman, F., Bass, L., Garlan, D., Ivers, J., Little, R., Nord, R., Stafford, J. 2003. Documenting Software Architectures: Views and Beyond¸ Addison-Wesley, Boston.

Gyselinck, V., Tardieu, H., 1998. The Role of Illustrations in Text Comprehension: What, When, for Whom, and Why?. In: van Oostendorp, H, Goldman, S. R., (Eds.) The Construction of Mental Representations During Reading, Lea, ISBN 0-8058-2429-4

Hilliard, R., 2001. Viewpoint modeling. In: Proceedings of 1st ICSE Workshop on Describing Software Architecture with UML.

IEEE, 2000. IEEE recommended practice for architecture description, IEEE, IEEE Standard 1471.

Jonkers, H. (ed.), de Boer, F., Bonsangue, M., van Buuren, R., Groenewegen, L., Hoppenbrouwers, S., Lankhorst, M.M., Proper, H.A., Stam, A., van der Torre, L., Veldhuijzen van Zanten, G., 2003. Concepts for Architectural Description, Telematica Instituut, Enschede, The Netherlands, http://archimate.telin.nl

Koning, H., 2003. Method to enhance the stakeholder orientation while designing viewpoints, request for comments, see http://www.cs.vu.nl/~henk/index.php?page=2003

Koning, H., Dormann, C., van Vliet, H., 2002. Practical Guidelines for the Readability of IT-architecture Diagrams. In: Proceedings of the 20th Annual International Conference on Documentation, ACM SIGDOC 2002, Toronto, October 2002, pp 90-99.

Koning, H., van Vliet, H., 2004. Real-life IT architecture design reports and their relation to IEEE Std 1471 stakeholders and concerns, submitted for publication.

Kotonya, G., Sommerville, I., 1998. Requirements Engineering, John Wiley & Sons, Chichester, ISBN 0 471 97208 8

Kruchten, P.B., 1995. The 4+1 view model of architecture, IEEE Software **12** (6) 42-50.

Parnas, D.L. & Clements, P.C., 1986. A Rational Design Process: How and Why to Fake It., IEEE Transactions on Software Engineering, 12, 2, 251-257.

Soni, D., Nord, R.L., Hofmeister, C., 1995. Software architecture in industrial applications. In: Proceedings 17th International Conference on Software Engineering, ACM, pp 196-207.

van Vliet, H., 2000. Software Engineering: Principles and Practice, John Wiley & Sons, Chichester.

M. van Welie, 2001. Task-based User Interface Design, Ph.D. Thesis, Vrije Universiteit, Amsterdam, The Netherlands, http://www.cs.vu.nl/~martijn

Henk Koning has over 20 years practical experience in system development and architecture. This article is part of his Ph.D. study on "Communication of IT-architecture".

Hans van Vliet is a Professor in Software Engineering at the Vrije Universiteit. His research interests include software architecture and software measurement. Before joining the Vrije Universiteit, he worked as a researcher at the Centrum voor Wiskunde en Informatica (Amsterdam) and he spent a year as a visiting researcher at the IBM Almaden Research Center in San Jose, California. Hans has an M.Sc. in Computer Science from the Vrije Universiteit and a Ph.D. in Computer Science form the University of Amsterdam.

5 Inquiry Tool for Stakeholder Concerns of Architectural Viewpoints: a Case Study at a Large Financial Service Provider

Henk Koning, Rik Bos, Sjaak Brinkkemper

Department of Information and Computing Sciences, University of Utrecht, P.O.Box 80.089, 3508 TB Utrecht, the Netherlands.
Emails {h.koning, rik, s.brinkkemper}@cs.uu.nl.

Abstract

IEEE Std 1471 defines architectural views as addressing related stakeholder concerns, but gives no guidance as to what architectural concerns are or how they should be found. In this paper we present an inquiry tool to solicit IEEE Std 1471 stakeholder concerns. The tool is a list of interview questions which help a stakeholder to express his concerns. The tool helps the enterprise architect to be aware of which architectural concerns are relevant in a given situation. We demonstrate the use of this tool in a case study at a Dutch bank, where an architecture documentation practice is evaluated and three stakeholder concerns are uncovered that are not yet addressed.

5.1 Architectural concerns of stakeholders

Architecture is a relatively new branch of study within software engineering. IEEE Std 1471 [4] defines it as "Architecture is 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". Reference [10] places the architecture definition phase in the software life cycle between the requirements engineering and design phases. In this phase the interests and concerns of all stakeholders are taken into account to come to a well-balanced solution. The setup of the documentation of architecture should reflect these concerns. According to IEEE Std 1471 the setup consists of a number of views, each of which address a set of related stakeholder concerns. IEEE Std 1471 gives no guidance as to which views should be present, or what architectural concerns are or how they should be found. To fill in this gap we have devised a method to define architectural views. The first step of our method deals primarily with eliciting the stakeholder concerns. In this paper we report on this activity and we present a generally usable inquiry tool we have devised for it, which consists of a list of interview questions.

The organization of this paper is as follows: in the rest of section 1 we introduce IEEE Std 1471, we introduce shortly our method viewpoints design and we describe related work. In section 2 we mention the research approach and describe the company that was involved. We present a prediction of stakeholder concerns and give a description of the interview questions. In section 3, the main section of this paper, we show a selection of the interview results. In section 4 we draw our conclusions. Section 5 summarizes future work. The appendix contains the list of interview questions.

TEAR2006 workshop of the 10th IEEE International Enterprise Distributed Object Computing Conference (EDOC 2006) ,16-20 October 2006, Hong Kong, China.

5.1.1 IEEE Std 1471

In 2000, the IEEE Standard 1471 [4] proposed a model of an architecture description and its context. It offers a high level conceptual model for architecture descriptions with explicit attention to the concerns of the stakeholders. For defining IEEE Std 1471 views it is important to have a good understanding of the stakeholder concerns.

An *architecture description* consists of *views* that are each made according to a *viewpoint*. A view is "A representation of a whole system from the perspective of a related set of concerns" (p.9), and a viewpoint is "A specification of the conventions for constructing and using a view. A pattern or template from which to develop individual views by establishing the purposes and audience for a view and the techniques for its creation and analysis." (p. 10). A viewpoint can be filed as *library viewpoint* for later reuse.

5.1.2 Method for viewpoint design

In a previous paper [6] we outlined a method for defining IEEE Std 1471 viewpoints. The method presumes that at least an outline of a non-IEEE Std 1471 compliant architecture description is present.

The method has four steps:

1. compile stakeholder profiles (a short table with: goal of the stakeholder, his tasks, his concepts, and his architectural concerns)
2. compile summary of architecture, in text (5 to 10 main statements) and graphics (model of key architectural concepts)
3. relate summary to concerns
4. define viewpoints

One of the points in the feedback we received on this method was the desire of practicing IT-architects to work from IEEE Std 1471 library viewpoints, instead of creating viewpoints from scratch. In fact they wished not to apply the method themselves but to have it done for them. We agreed to fulfill this wish and perform the method for one department, which had uttered this wish. We started with step 1, stakeholder profiles, especially with establishing stakeholder concerns. This paper reflects specifically on that activity. Stakeholder concerns are essential in the IEEE Std 1471 model and play a key role in our method for viewpoint design.

5.1.3 Related work

Greefhorst, Koning and Van Vliet [3] have created an overview of existing architecture frameworks. One could derive from those frameworks many possible viewpoints to use in documenting IT-architecture. An evaluation activity, comparable to the round of interviews discussed in this paper, would be necessary to establish the need for viewpoints.

Reference [2] offers many models and guidelines for composing a software architecture description. These models may fit a context for the viewpoints we are interested in.

Smolander and Päivärinta [9] have examined the reasons for making architecture descriptions in practice. Interviews with various stakeholders of architecture in three companies showed that beside the traditional use as a starting point for system design, architecture documents serve to communicate, to negotiate and to capture knowledge. These reasons can be seen as stakeholder concerns.

The elicitation of stakeholder concerns is in our view comparable to the elicitation of requirements for system development for which [8] has outlined processes and techniques. He advises open interviews, as well as interviews structured by a predefined list of questions.

5.2 Research setting

In this section we lay out the research setting. We first describe our project approach (research method, participating company, interviewees). Then we present stakeholder concerns that were compiled as a prediction. We finally introduce the interview questions.

5.2.1 Project approach

This research project follows an “action research” approach [1]. Action research is applied because it is here not possible or not viable to mimic a real life situation in a laboratory environment. In action research complex social processes are studied by introducing changes into these processes and observing the effects of these changes. Knowledge obtained through the use of this approach is difficult to validate in terms of the natural science view.

In an action research setting the researcher is actively involved in the practice, with expected benefit for both researcher and practitioners. The researcher works as an insider in order to understand and discern the issues related to the subject matter. The knowledge obtained can be immediately applied; there is not the sense of the detached observer. The research is a process linking theory and practice. We follow the interpretive stance of action research. In action research five cyclic steps are defined: diagnosing, action planning, action taking, evaluation, specifying learning. The evaluation of our method for defining IEEE Std 1471 viewpoints has resulted in learned lessons, of which one is that architects prefer not to define viewpoints themselves, and has brought us to a new diagnosing activity by means of these interviews. Based on the interviews we will do action planning in the form of proposing new viewpoints.

The action research participants were architecture interested stakeholders working with a financial services company we call FSC in this paper. FSC is a Dutch international bank that attaches great importance to IT architecture to manage its very complex IT operations. With the introduction of FSA (Financial Services Architecture) a major overhaul of all information systems is underway at FSC.

Our contacts at FSC are with an architecture department that maintains a database with information on hundreds of applications (information systems) and their relations. They are often asked to create diagrams of the applications that support a certain business domain. An architecture study, in their case, goes a step further and outlines a 'to be' situation for a business domain. That is to say, which (reduced) set of applications should support the business domain in the future and what software needs to be bought or built. So for this type of reports the concerns are needed.

The eight architecture interested stakeholders we interviewed were “customers” of the department. Six interviewees were part of the IT department at FSC, in the three roles: manager system development (called Dev1 and Dev2 in this paper), generally interested (GI1 and GI2) and Chief Information Officer (CIO1 and CIO2). Two interviewees were managers from an FSC business department (Biz1 and Biz2). The department considers this a good representation of their customers. Except for CIO1 all interviews were recorded in Dutch, translations in this paper into English are by us.

The department uses a report template of which this is the table of contents:

- Management summary
- Assignment
- Business architecture (organizational structure, functional breakdown, process models, principles)
- Logical IT architecture (ideal positioning in FSA)
- Current Architecture (implementation models, IT solutions)
- Target IT Architecture (future implementation models)
- Migration

We take the table of contents as a starting point. Our research question in this paper is: what are, for this department at FSC and their type of documents, the concerns of the stakeholders and which concerns are not addressed by the current architecture documentation practice?

5.2.2 Forecast

A forecast was prepared in cooperation with the department. We wanted to be explicit about what we expected and create a reference point for evaluating the outcome of the interviews. See.Table 5-1.

Table 5-1 Predicted concerns per category of interviewees

Category	Concerns
Dev	Which business requirements do I have to fulfill?
	To which architectural guidelines do I have to adhere?
	Does my design fit within the IT architecture?
GI	Is there some common ground between this architecture and my interests?
	If so, how can I react (if necessary)?
CIO	How to get a minimum number of implementations over the regions?
	How can I minimize the costs?
	How can I get more control over the projects?
	How can I increase the speed of delivery?
Biz	Do I get the right IT systems for my department(s)?
	Are the costs of the IT systems not too high?
	How can I make the most of my opportunities using the IT possibilities?
	Do I get sufficient service (from IT-department)?

5.2.3 The interviews

To make the results of the interviews comparable a detailed list of questions was prepared, see the Appendix.

The list of questions covered five topics:

1. the perception of the interviewee of his own role/function in the company (14 questions).
2. the perception of the interviewee of information technology (17 questions).
3. the ideas of the interviewee on business reports in general (4 questions).
4. the wishes of the interviewee concerning future IT-architecture reports in general (5 questions).
5. the comments of the interviewee on a specific example of an IT-architecture report (10 questions).

The interview questions have a broad focus, broader than strict ‘architecture’, to increase the chance that the interviewee expresses his real concerns, the issues that really motivate him/her.

Some of the questions in topic 1 ask for attributes of the so called stakeholder profile: goal, tasks, concepts, concerns. This is part of our method for defining IEEE Std 1471 viewpoints. Concerns play a central role and come back in all the topics.

Section 5 of the interview relates to an existing report. The department chose for this an architecture study of future systems of the domain called Market Risk. This report is representative for the kind of reports they produce.

Question 5.3 'Here is an outline of this reports, could you please indicate for each of your main questions where in the report you find the answer to the question? (this can be done after the interview)' is reminiscent of our research in which we scanned four existing reports for the relation of the content to the concerns of the stakeholders. See [5].

5.3 Interview results

The document with the interview questions and all the answers of the interviewed stakeholders spans 47 pages. For the sake of brevity we limit ourselves in this paper to the highlights, in particular to the questions that proved very effective in soliciting concerns: 1.6, 2.12, 4.1 and 5.2. The other questions still provided useful information for the architecture department, for instance about the communication preferences of the stakeholders, but we will not delve into that in this paper. A number of questions proved to be not very helpful. We have indicated these in the appendix with a '*' and they will be omitted from a future version of our tool.

We now present a few tables with answers to interview questions. Italics in the answers are by us. After the answers we give a short comment.

5.3.1 Concerns - general

Question 1.6 targets concerns at the level of the function in general of the interviewee. See Table 5-2. The interesting point here is 'to which concern can the architect contribute?' The italics indicate concerns that, in our view, can be met by a good working architecture department.

Table 5-2. Answers to 1.6: What are your worries? What things did you run after, yesterday? Today? Tomorrow?

Interviewee	Answer to interview question
Dev1	Outsourcing / off shoring (to India) *Successful completion of large scale projects* Too many meetings (too low productivity) Attitude of employees (lot of talk about rights, little attention to plights) Huge hierarchy, much overhead (too many chiefs, too few indians)
Dev2	Alternative question: what keeps your mind busy? Answer: *direction of the bank with regard to it, what are we going to do in the coming years? Which systems to use or not*; how to organize the big changes (outdated applications; developers of age… while technology is changing strongly)(few days training is insufficient, big conceptual changes; motivation? Ability to change?)
GI1	Budget wise (for instance medium term planning coming year, safeguard architecture capacity, functional CIO supports from a distance) Planning wise: little is done with an eye on the long term and that makes it difficult for architectures. *Projects respond to short term and push aside long term goals.*
GI2	Having been outsourced, we are divided even further, which makes *knowledge being lost*. Difficulty of NL-organization with regard to international financial markets. Budgets don't coincide.

Inter-viewee	Answer to interview question
CIO1	*Difficulty of change management*, continental Europe not completely ready for the change. Local vision may abide. *Legacy governance*, old habits.
CIO2	*Greatest worry is that we have become so big and complex, and yet want to act quickly*. The direction of the reorganizations is ok, but the implementation is not manageable (interviewee is responsible for all supporting systems). Lack of adequate governance and insight in projects portfolio (impact on each other). The people, busy outsourcing a substantial part of work. 20% less people in 1 ½ year? The work changes from carrying out to directing. There is a low turnover, so a lot of people are older than 40.
Biz1	(Alternative because of coming end of function: what were your worries?) *Keeping projects in control* (scope, goal, end date; will water down if interviewee stops). Prevent friction. There are so many opportunities that are not seized. One goes on and on in the present ways. Innovation will diminish in the new organization. Functional people are not much IT-minded. Ad hoc driven.
Biz2	See to it that the support functions (Risk management, finance & control, operations, IT, HR) understand what the Front Office (FO) needs, worldwide in a uniform way. See to it that priorities are understood. This means a lot of communicating. Tomorrow: *see to it that it-solutions are aligned and fully leveraged in line with FO requirements*.

All stakeholders summed up a small list of worries. Keeping control is a recurring theme. Architecture enforces the control over large scale projects, gives direction with regard to which systems to use or not, enables change, highlights the essential needs of user groups (gives understanding and promotes alignment), captures essential knowledge, etc. Quite a few stakeholders mention people problems: wrong attitude, inability to change, lack of vision, job losses, and lack of understanding.

5.3.2 Concerns – IT

Question 2.12 asks for the weak points in IT. Indirectly it inquires for the IT-concerns in general of the interviewee. See Table 5-3. The possible contribution of the architect here seems to lie with two topics: tensions with the business and the big amount of legacy systems. With italics we want to draw your attention to these topics.

Table 5-3. Answers to 2.12: What do you see as the weak points of the way information technology is used in your company?

Inter-viewee	Answer to interview question
Dev1	Too much internal focus, Bureaucratic/ paper-work (few functions automated)
Dev2	We drag along *a lot of history*, cumbersome to get rid off (technical and organizational (users and it-persons)) What – how discussion: distinction sometimes difficult, sometimes unwanted. *End users want specific IT-solution which leads to stalemate.*
GI1	*Seen from the business: they use IT as a short term resource. This leads to ad-hoc solutions which are not so beautiful and with which you will not reach the long term goals.*
GI2	Complexity (applications). Infrastructure is at places outdated.
CIO1	We have *too much legacy*, ergo invested too much. We moved not quickly enough to new systems, new platforms. So we have too much ad hoc (bespoke) development.
CIO2	*Too much diversity*.

Inter-viewee	Answer to interview question
	Tension between project interests from business vs. architecture interests. Each time again negotiating.
Biz1	Unfounded trust (too easily it is assumed that things will go right). Vulnerability, unpredictability.
Biz2	One time *60 different labels* in the market, each own IT infrastructure.

From this we determine two interesting concerns. First, there is three times a concern uttered regarding tension with the business. Good quality architecture documents can be instruments in easing this tension. The second one that stands out is the big amount of legacy systems / too much diversity. The department is already aware of this last concern and addressing it in their publications by proposing reductions of the number of systems.

5.3.3 Concerns - architecture

Question 4.1 asks the concerns for IT-architecture in general, see Table 5-4. The italics indicate concerns that are not met by the current documentation practice.

Table 5-4. Answers to 4.1: Suppose one of these days a new project is launched and an architecture study is started to lay the foundation. What questions would you like to see answered by the study?

Inter-viewee	Answer to interview question
Dev1	Necessity of change. Goal, scope, project results, project organization and composition (structure project organization), way of doing, *time lines*, risks, *costs/benefits*, principal solution.
Dev2	Demarcation of domain: which enterprise functions, which applications. What often is missing: being conscious of how you come to a choice, what makes an architecture less good or bad? For instance “application must be used for all products” (is often not possible, remains unnoticed). How big is a system allowed to be? For how many products/systems? Which parameters form the boundaries?
GI1	What is it about? Which information plays a role, and from which domains? How do I expect the applications to look (which functions are in it)? What is in it and what is the consistency with the environment? How are the links realized? *What is the targeted infrastructure?*
GI2	How does it connect to the group architecture (standards)? Does it indeed connect?
CIO1	Simply: description of business requirements + high level how is the business need resolved + *how does it fit in the rest of the architectures.*
CIO2	New project, first question: what is the area you are touching (current destination plan), do I have common interests, are there *other projects in that area*, what is the impact? (an architecture report goes from business to system plan & infrastructure)
Biz1	Where does that change touch me, what is the impact? Red and green colors … Is it complete? Is everything covered? For instance, are all applications mentioned?
Biz2	To what degree support the current applications the business processes? Which resources are there for support and which connect in the best way?

Practically all of these points are architecturally interesting. The main points are: scope, what business functions are supported by which applications, what data is involved, how does it relate to other architectures or projects, what infrastructure is needed?

Various answers, but on the whole they are very supportive of the current architecture model of the department. Some extra concerns identified are:

infrastructure, time lines, cost/benefit, relation to other architectures and other projects, see *italics*. As we will see, time and money are a recurring theme.

5.3.4 Concerns – specific report

Question 5.2 asks for concerns at the level of one specific report, see Table 5-5. In these answers we look for topics that are not covered in the present documentation setup.

Table 5-5. Answers to 5.2: Suppose you would receive this report today for the first time, for which questions would you seek answer when you would start reading?

Inter-viewee	Answer to interview question
Dev1	What is the necessity of change? Who did collaborate? How will the migration go, and what are the consecutive steps? I miss the context. 67 slides is very much; slides with too many words. Wish: first essence, main principles; details in appendices.
Dev2	*Who has asked for it?* Who will decide over the recommendations? Which choices have been made?
GI1	*Why did I receive it?* What are the consequences for my domains? Does it fit in my idea of market risk?
GI2	*Why did you send it?* With what goal? How does it connect to the group architecture (standards)? Does it connect indeed?
CIO1	How is the business operating? How does it want to operate? How will the needs be solved. This is a catch-up document.
CIO2	What are the subject and the scope? Position in the destination plan. What will the migration look like (80% costs)? *What does it mean in time and labor and can we cope with that?*
Biz1	(See it today for the first time) What is it about? Can I find quickly what the intention is? *Is it indeed meant for me?* Do I have to read it? (Comment: not clear right away …) Is it still valid? …….. is quite old. Are the sticking points of the current situation described? Which principal solution is being proposed?
Biz2	Can I deduce from the report whether the Market Risk supporting systems give good support to the Front Office (where the risk has to be managed)? What do we have today and in which direction are we heading?

What strikes us is that there are quite a lot very down to earth questions: why read it? Who has asked for it? Who will decide over it? Who has worked on it? We label this as the organizational context of the project. The department is used to address these issues in separate accompanying letters. Other, architectural aspects mentioned are: How is the business operating? How does it relate to other architectures or the destination plan? What are the sticking points that are solved? What will the migration look like? These are covered by the present document setup, so this is affirmative for the work of the department. Only not covered is the question 'What does it mean in *time* and *labor* and can we cope with that?' (CIO2).

5.3.5 Some miscellaneous questions

In this section we want to shortly mention a few questions that have not delivered many concerns, but which did gave us pointers for our conclusions.

1.5. What are the 'things' that make up the content of your work? Examples: money, employees, products, …

Two subjects were mentioned six times: *people* and *money*. Meetings were mentioned four times, and information systems and architecture documentation three. Other things were mentioned once or twice.

2.7: What are the 'things' IT-architecture reports should deal with?

Question 2.7 asks for the interviewee's grasp of essential architectural concepts, which is connected to the stakeholder concepts of the stakeholder profile (attribute 3). The topics mentioned by the stakeholders in answer to this question are very much in line with the architecture definition of the department, as shown by the existing report. So this is also affirmative of the work of the department. Not in line was one utterance 'The *costs*, *time* aspects, capacity aspects.'

2.13. What is your definition of a failed IT-project?

The interviewees gave a unanimous response. A failed project runs out of time or money, or it delivers no agreed functional results. This stresses again the importance of *time* and *money* in the company culture of FSC.

3.2. Is there a business report that you see as a bad example, that shows a way of communicating that should be avoided? If so, for what reasons?

Most stakeholders used this question to utter wishes for good reports. One of the wishes is: provide cost/benefit analysis.

5.4 Discussion and concluding remarks: architectural concerns

In the context of the architecture department of FSC our research questions are: what are the concerns of the stakeholders and which concerns are not addressed by the current architecture documentation practice? In chapter 5.3 the concerns of the stakeholders are exposed, as found by means of the interviews. In this chapter we summarize these, but lay emphasis on the concerns found that are not addressed by the current documentation practice and present some other findings.

Our main conclusion is that the present setup of an architecture report addresses most concerns the stakeholders have for this kind of reports, but that additional viewpoints are necessary. The answers of the interviewees were affirmative for the present work of the architecture department. Another finding is that our method for viewpoint design needs to be extended.

If we summarize the answers of the interviewee's to our list of questions, we can state that the stakeholders have these concerns that are covered by the present document setup: how can we explain IT to the business? How can we retain control over large projects? Which systems are we to use or not? What are the essential needs of the business? How to reduce the big amount of legacy systems? What business functions are supported by which applications? What is the relation to other architectures? What are the sticking points that are solved? What will the migration look like?

The additional concerns are related to *time* aspects, *money* aspects and *people* aspects. These aspects show prominent in the answers of the interviewees, but are not covered by the existing architecture document setup of the department. These aspects have a very general nature and, as best practice, can be part of any architecture documentation.

Time aspects entail questions like when will which parts of the proposed architecture be realised? And how much time is accounted for in the migration. Many things are happening at the same time at FSC and proposed activities are not taken seriously when they are not fixed on the calendar somewhere. The department had until now the habit to leave the fixing of dates to the next process after the publication of an architecture study.

Money aspects entail questions like how much do the current operations cost? How much are the costs of the migration? How much will the operations cost after that? One of the big business drivers at FSC at the moment is the reduction of operating costs. For each new activity it is important to know what effect it will have on operating costs. Until now the precise calculation of financial benefits and burdens was left to the system design projects that would follow on the architecture study. It is the intention of the department to finish up architecture reports with financial data with the help of a portfolio manager.

People aspects entail questions like: how much more or less people will work at FSC after the introduction of the proposed architecture? What new skills are demanded of system developers or end users? Will the migration activities be conducted by FSC personnel or will they be outsourced? This is a new area of attention.

The adaptation of our viewpoint design method entails the extension of step 1 (compile stakeholder profiles) with an activity to perform a stakeholder interview, based on the interview questions in the Appendix. The big difference between the forecasted concerns and the found concerns makes this a necessary extension.

We found the round of interviews a meaningful exercise and a good way to evaluate the current documentation practice of the department. Some needed additions to the documentation setup were found.

The list of interview questions functioned well as a ‘Stakeholder Concern Inquiry Tool’. Many architectural stakeholder concerns were uncovered and some relevant ingredients of the company culture were brought to our attention. The interview questions carry little or no reference to circumstances that are specific for FSC. We feel the questionnaire is widely applicable in situations were decisions need to be taken over which information systems should support a certain business domain. The open nature of most questions stimulates the stakeholder to express himself, but makes it also necessary to apply a filtering on the answers, as shown in this paper. Some questions that did not help in soliciting concerns and did not reveal any other useful evaluation information will be evicted from the list (indicated by a ‘*’ in the appendix). What are left are 24 questions to reveal concerns of architectural descriptions. The list of questions can be used to get a better grip on what concerns are relevant in a certain situation. In that sense it fills a gap in IEEE Std 1471.

5.5 Future work

We want to turn the list of questions into a standard questionnaire that can be used to evaluate an architecture documentation practice. Some questions that were not so helpful in soliciting stakeholder concerns will be removed from the list. The questionnaire will be made available on-line [7].

We want to see what more can be learned from practices in the area of requirements engineering and apply that to soliciting of stakeholder concerns.

5.6 Acknowledgements

We thank Rich Hilliard and Slinger Jansen and anonymous reviewers of an earlier version of this article for their valuable comments. We thank FSC for giving us the possibility to conduct these interviews.

5.7 References

[1] R.Baskerville, Investigating Information Systems with Action Research, Communications of the Association for Information Systems, 2(19): 1- 31, 1999.

[2] P. Clements, F. Bachman, L. Bass, D. Garlan, J. Ivers, R. Little, R. Nord, J. Stafford, Documenting Software Architectures: Views and Beyond¸ Addison-Wesley, Boston, 2003.

[3] D. Greefhorst, H. Koning, H. van Vliet, The many faces of architectural descriptions, journal Information Systems Frontiers (vol. 7, no. 3), 2006.

[4] IEEE, IEEE recommended practice for architecture description, IEEE Standard 1471, 2000.

[5] H. Koning, H. van Vliet, Real-life IT architecture design reports and their relation to IEEE Std 1471 stakeholders and concerns. Automated Software Engineering (ASE) Journal (13:2), 2006.

[6] H. Koning, H. van Vliet, A Method For Defining IEEE Std 1471 Viewpoints. Journal of Systems and Software, volume 79 no 1, 2006.

[7] Homepage of this research: http://www.cs.uu.nl/people/koningh.

[8] G. Kotonya, I. Sommerville, Requirements Engineering: Processes and Techniques, John Wiley & Sons, 1998.

[9] K. Smolander, T. Päivärinta, Practical Rationale for Describing Software Architecture, Beyond Programming-in-The-Large, proceedings of 3rd Working IEEE/IFIP Conference on Software Architecture (WICSA3), August 25-30, 2002, Montreal, Québec, Canada, Jan Bosch et.al., ISBN 1-4020-7176-0, page 113-125.

[10] H. van Vliet, Software Engineering: Principles and Practice, John Wiley & Sons, Chichester, 2000.

5.8 Appendix Interview questions

Stakeholder Concern Inquiry Tool

The list of interview questions that were used in the case study treated in the paper. With a '*' are questions marked that will be omitted from future versions of this tool.

1. Perspective of interviewee on his own role in the company

1.1. What is the name of your role/function in the company?
1.2. What is the goal of your role/function?
1.3. What are your tasks?
1.4. * Are there tasks that you could perform, but which you consider outside your responsibility?
1.5. What are the 'things' that make up the content of your work? Examples: money, employees, products, …
1.6. What are your worries? What things did you run after, yesterday? Today? Tomorrow?
1.7. * How do you keep control over your work?
1.8. * What kind of happenings can derail the functioning of your department?
1.9. * How do you take part in the decision taking processes in the company?
1.10. * From which parties do you receive directives for your work?
1.11. * To which parties do you give directives?
1.12. * What are the reporting guidelines to which you must adhere?
1.13. * Do you have salary bonus system, and if so, what are the mechanics of that?
1.14. * What are your plans for the future?

2. Experiences/expectations/relevancy of strategic IT projects

2.1. What does IT mean to you?
2.2. * What effect does IT have on your department?
2.3. How would you describe what a 'strategic IT decision' is?
2.4. Can you give some examples of strategic IT decisions?
2.5. * Are you involved in decision taking regarding IT? If so, how?
2.6. What would be your description of IT-architecture?
2.7. What are the 'things' IT-architecture reports should deal with?
2.8. * What is redundant in an IT-architecture?
2.9. In what circumstances is an architecture study needed?
2.10. * In what circumstance is an architecture study superfluous?
2.11. * What do you see as the strong points of the way information technology is used in your company?
2.12. What do you see as the weak points of the way information technology is used in your company?
2.13. What is your definition of a failed IT-project?

2.14. * What is your definition of a successful IT-project?
2.15. In which ways do you receive information about strategic IT-projects?
2.16. What do you like of each way you receive information about strategic IT-projects, what do you dislike, and why?
2.17. Do you have a wish for the way in which you would like to receive information about strategic IT-projects in future?

3. Business reports in general

3.1. * Is there a business report that you would like to be used as an example for future reports? If so, for what reasons?
3.2. * Is there a business report that you see as a bad example, that shows a way of communicating that should be avoided? If so, for what reasons?
3.3. * How do you compare written reports on paper to information on intranet webpages or powerpoint slide shows? Advantages/disadvantages? Application?
3.4. * Would you like to have templates used all the time?

4. Wishes future IT-architecture reports

4.1. suppose one of these days a new project is launched and an architecture study is started to lay the foundation. What questions would you like to see answered by the study?
4.2. * What questions should be skipped in an architecture study?
4.3. * If you could fully trust the IT-department, would that change the way you want to be informed?
4.4. At what moments would you like to be involved in the architecture study? For what issues/reasons?
4.5. Wishes with regard to publication media and form?

5. Opinions/wishes about an example report

5.1. Can you please give your own description of what this report is about?
5.2. Suppose you would receive this report today for the first time, for which questions would you seek answer when you would start reading?
5.3. * Here is an outline of this reports, could you please indicate for each of your main questions where in the report you find the answer to the question? (this can be done after the interview)
5.4. * Are you missing information in the report to answer your questions?
5.5. * If you had to explain the content of this report to a newcomer, to which section (text or diagram) would you refer?
5.6. * How much time did you spend on reading?
5.7. * How quick could you find the information you needed?
5.8. Do you consider this report good readable? Can you explain?
5.9. Do you have any other comment on the setup of this document? (Please don’t comment on the content, but only on the way it is presented.)
5.10. Do you agree or disagree with the report? To what extent? Advantages/benefits? Disadvantages/losses?

6 The many faces of architectural descriptions

Danny Greefhorst
IBM Business Consulting Services
The Netherlands
greefhorst@nl.ibm.com

Henk Koning
Freelance researcher
The Netherlands
koning@arc-it.nl

Hans van Vliet
Vrije Universiteit, Amsterdam
The Netherlands
hans@cs.vu.nl

In recent years architecture has acquired recognition as playing a pivotal role in change processes. Despite this recognition, describing architecture has proven to be difficult. Architecture frameworks have been defined to address this problem. However, there are many of them, and together they leave us with seemingly contradicting terminology. What are the underlying forces that caused people to create so many different frameworks? What do these frameworks teach us about the essence of architecting? Where do I start to select or create a framework for my current project? With these questions in mind we set out to perform a comparison of existing architecture frameworks. We ended up with a deeper understanding of the function of a framework, and "discovered" nine fundamental dimensions that seem to underlie architectural thinking. These "base dimensions" can be used to clarify the meaning of individual architecture documents independent of the framework they originate from, and they can be helpful in defining new architecture frameworks or situational architecture descriptions. In this paper we also relate our findings to IEEE 1471, which is another important generalisation of existing frameworks.

6.1 Introduction

Architecture in IT has gained acceptance as a means to guide IT change processes. Although people tend to disagree on the exact definition, architecture can be seen as the high-level structure of a system. It describes fundamental aspects of the system, and guides the persons that actually design and build the system. Architecture needs to be described in a document: an architectural description. Also, the architectural description needs to be structured into manageable "chunks" that each addresses a number of aspects of the architecture. There is no universal agreement on the "chunks" that an architectural description should consist of. Architecture frameworks should provide guidance in this area, but the problem is that there are so many of them. Even when there is agreement on the use of a framework, the peculiarities of a specific project often make it necessary to deviate from the framework. IT-architecture consultants who work in varying circumstances have to spend extra time to get acquainted with local templates, and it may take some time before all the meanings are clearly understood. The trade of architecting is visually manifested mainly by the frameworks. They are the signs of mastery achieved. Persons who are new to the field, like junior architects, and who see so many diverging frameworks, ask themselves "what is going on here?". Can you imagine a doctor in a hospital saying "I have 18 ways to record your case in my files".

These observations motivated us to investigate frameworks for architectural descriptions, and try to discover their fundamental structure. Architecture frameworks order architectural descriptions along one or more axes, and typically visualize the resulting architectural space spanned by these axes. A cell in this n-dimensional space denotes an architectural description that corresponds to the characteristics of the accompanying column and row. We call these axes "dimensions", and call specific columns or rows in these dimensions "values". Further analysis of these dimensions led to the identification of nine "base dimensions", that are the foundation for the dimensions found in architecture frameworks. We will

The original publication is available at www.springerlink.com, see http://dx.doi.org/10.1007/s10796-006-7975-x.

describe these base dimensions, and illustrate their relationship to the dimensions as they occur in existing architecture frameworks. The latter are typically a combination of one or more “base dimensions”.

The main contribution of this paper is the insight that architectural “dimensions” should be made explicit, and are based on a number of “base dimensions”. It is not our intention to introduce a new framework. Although we studied quite a number of frameworks, we do not contend that our list of base dimensions is complete. Also, the values within the base dimensions presented are merely described to illustrate the base dimensions. Finally, we did not strive to make a complete survey of architecture frameworks, but only to have a solid enough basis for analyzing the “logic of architectural frameworks”. Readers are urged to use the base dimensions presented as a reference point to position individual architecture documents or to better understand the essentials of existing architecture frameworks.

An important milestone in the field of architecture descriptions is ANSI/IEEE Std 1471 [IEEE1471], which was published in 2000. We will refer to this standard as IEEE 1471. IEEE 1471 proposes to structure architecture descriptions in views which are directly related to stakeholder concerns. In this paper we point to some strengths and weaknesses of this approach and we show that our findings are complementary to IEEE 1471.

This paper is organised in three sections. In the first section we describe the current situation of the architecture frameworks. We will give a short description of two architecture frameworks, just to introduce the notion of an architecture framework, and the concept of “dimension” to readers unfamiliar with them. We then list a number of architecture frameworks, and the dimensions we discovered in them. This is followed by an analysis, leading to some general observations and essentials of architecting. In the second section we elaborate on the concept of “dimension”, and propose a list of base dimensions in architecture. We illustrate the usage of the base dimensions with an example. In the third section we relate our work to IEEE 1471. We conclude with a short recap and acknowledgements and references.

6.2 Architecture frameworks

Architecture frameworks offer a standard approach to architecture. This approach may encompass a model for architectural descriptions, as well as a method to produce them. Some architecture frameworks focus on the architectural descriptions, while others focus on the method. In this paper we are mainly interested in the way architecture frameworks approach architectural descriptions, and structure them into one or more dimensions. Further analysis of the space of architecture frameworks shows that they can be divided into two categories: enterprise-class frameworks and application-class frameworks.

Enterprise-class frameworks are aimed at business units, complete organisations or even industry sectors. These frameworks often have multiple dimensions, potentially leading to a large number of architectural models. An enterprise architectural information base may contain many separately maintained documents. Examples of enterprise-class frameworks are the Zachman Framework for Information Systems Architecture (ISA) [Zachman87, Sowa92], the Information Framework (IFW) [Evernden96], The Open Group Architecture Framework (TOGAF) [OpenGroup03], Integrated Architecture Framework (IAF) [Goedvolk99] and Methodology for Architecture Description (MAD) [Meinema99].

Application-class frameworks describe the architecture of a specific (software) application or a group of similar applications, and typically comprise a small number of architectural models. The information in application-class frameworks is often more fine-grained than the information in enterprise-class frameworks. Well-known application-class frameworks are the 4+1 model [Kruchten95], the framework of Siemens [Hofmeister95][Hofmeister00], and the 2+2 model of the Vrije Universiteit [Lassing01].

The following paragraph describes the Zachman framework and the 4+1 model in more detail as typical examples of enterprise-class and application-class frameworks, respectively. It also illustrates the concept of dimension.

6.2.1 Showcases

6.2.1.1 Zachman

The foundation for enterprise-class frameworks was laid by John Zachman in his 1987 article [Zachman87] in which he describes a framework for the architecture of information systems. His idea was that architecture for information systems could be inspired by architecture in more mature engineering disciplines. He saw that the architectural models in these engineering disciplines showed a lot of similarities and could be combined in a generic model. Zachman recognised two dimensions: perspectives of specific target audiences and the types of architectural descriptions.

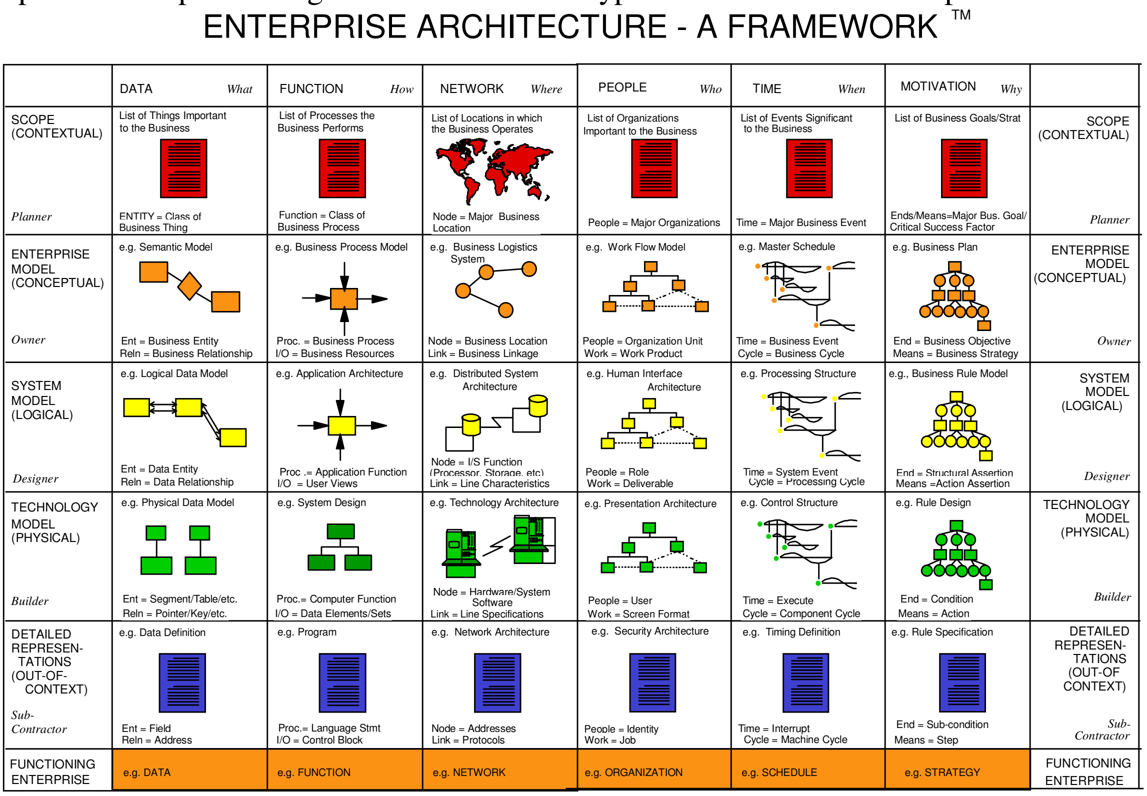


Figure 6-1 Zachman framework for Enterprise Architecture

Potential perspectives are those of: the planner, the owner, the designer, the builder and the subcontractor of an information system. Later on, Zachman gave these perspectives more logical names, and they were labelled the contextual, conceptual, logical, physical and out-of-context perspectives. The out-of-context perspective denotes that at this level parts are typically fabricated outside the larger context in which they are used.

The types of description dimension finds its origin in Zachman's observation that the same elementary questions of what, how, where, who, when and why can always be answered in different contexts. For information systems these questions are translated to data, function, location, people, time and motivation. The other observation was that both dimensions could vary independently, leading to 5*6 = 30 different kinds of architectural models for one information system. In the framework these models are depicted in a matrix with columns for the types of description and rows for the perspectives (see Figure 6-1).

6.2.1.2 4+1

A well-known application-class framework is the 4+1 model [Kruchten95] for software (see Figure 6-2).

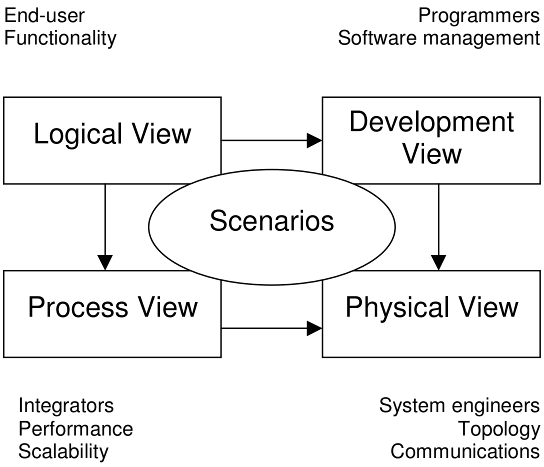


Figure 6-2 The 4+1 model

In contrast with the enterprise-class frameworks, this framework only has one dimension, which is not named explicitly. Like the Zachman framework, the views relate to different stakeholders and their concerns. There are four views, namely the logical, development, process and physical view. These views have a recognisable relationship with users (classes), developers (packages and files), integrators (processes, messages) and system engineers (nodes and networks). The fifth view contains scenarios that describe how the elements in the other views co-operate.

6.2.2 Overview

We now offer a summarized overview of architecture frameworks, and other architecture classifications we found (see Table 6-1). For each framework we list the source, the dimensions and the values in the dimensions. The dimensions are depicted as the rows next to the framework. A division of values into sub-values is shown in parentheses. The table will be the primary source of inspiration for our definition of dimension, and the base dimensions that we distinguish. Cells for dimensions and values that are empty indicate that the source does not explicitly name them.

Framework	Source	Dimension	Values
2+2 model	[Lassing01]		Context, Technical Infrastructure, Conceptual, Development
4+1 model	[Kruchten95]		Logical, Process, Development, Physical, Scenarios
ADS	[Youngs99]	Aspects	Functional, Operational

		Level	Specified, Physical
ARIS	[Scheer92]		Organizational, Data, Control, Function, Product/Service
Boar	[Boar98]		Infrastructure, Data, Applications, Organization
			Inventory, Principles, Models, Standards
CIMOSA	[ESPRIT93]	Instantiation	Generic, Partial, Particular
		Views	Function, Information, Resource, Organisation
		Derivation	Requirements Definition, Design Specification, Implementation Description
DYA	[Wagter01]		Business (Product, Process, Organisation), Information (Data, Application), Technical (Middleware, Platform, Network)
			Common Principles, Policies, Models
Evernden Eight	[Evernden02]	Types of information	
		Levels of understanding	
		Types of representation	
		Levels of transition	
		Types of knowledge	
		Levels of responsibility	
		Types of process	
		Meta levels	
Gartner	[Rosser02]	Scope	Multi enterprise Grid, Enterprise, Business Process, Brick
			Context, Concept, Logical
			Now, less than 2 years, 2 to 5 years
GEM	[Baat99]	Operational processes	External Infrastructure (Suppliers, Partners, Customers) Business Architecture (Business Organisation, Business Processes, Business Information), Application Architecture (Presentation, Business Logic, Data Access), Technical Architecture (Middleware, Operating System, Hardware)
		Migration Infrastructure	Operations & Support, Specification, Test, Training & Deployment Development & Maintenance Architecture & Engineering
GERAM	[IFIPIFAC98]	Life-Cycle	Identification, Concept, Requirements, Design, Implementation, Operation, Decommission
		Genericity	Generic, Partial, Particular
		Views	Entity Model Contents, Entity Purpose, Entity Implementation, Entity Physical Manifestation
GRAAL	[VanEck02]	Service Layers	Environment, Business mission and functions, Business processes, Software applications, Software platform, Processing and networking hardware
		Refinement	
		Lifecycle	Planning, Organizing, Directing, Controlling
		Aspects	Dictionary, Communication, Functions, Behavior, Quality
Herzum/Sims	[Herzum00]		Functional, Application, Technical, Project Management

IAF[3]	[Goedvolk99]	Main architecture areas	Business, Information, Information Systems, Technology Infrastructure
		Design phases	Contextual, Conceptual, Logical, Physical, Transformational
		Special viewpoints	Business and ICT System, Security, Governance
IFW	[Evernden96]	Types of information	Organization (Strategy, Structure, Skills), Business (Data, Function, Workflow, Solution), Technical (Interface, Network, Platform)
		Levels of constraint	Deconstruction (Domain Concept, Domain Classification), Composition (Generic Template, Design Context), Implementation (Operational Bound)
		Content	Organisation Model, Financial Services Data Model, Financial Services Function Model, Financial Services Workflow Model, DesignWare, Finance Industry Solutions, Technical Model, Financial Application Architecture
		Transformation	
		Ownership	Global, Industry, Enterprise, Local, Individual
		Route maps	
MAD	[Meinema99]		Inter-organizational, Organizational, Process, Information, Application, Distribution, Configuration
Maier/Rechtin	[Maier02]		Data, Behaviour, Form, Purpose, Performance, Managerial
March	[Hermans02]		Product, Process, Organisation, Information provisioning, Infrastructure
			Context, Concept, Logical
			Now, less than 2 years, 2 to 5 years
RM-ODP	[ISO10746]		Enterprise, Informational, Computational, Engineering, Technology
Siemens	[Hofmeister95] [Hofmeister00]		Conceptual, Module, Execution, Code
Tapscott	[Tapscott93]		Business, Work, Information, Application, Technology
TOGAF	[OpenGroup03]	Architecture Domains	Business, Data, Applications, Technology
		Architecture Continuum	Foundation, Common Systems, Industry, Organisation
Zachman	[Zachman87] [Sowa92]	Types of description	Data, Function, Network, People, Time, Motivation
		Perspectives	Contextual, Conceptual, Logical, Physical, Implementation, Out-of-Context

Table 6-1 Existing architecture frameworks

6.2.3 Observations

6.2.3.1 Confusion

An analysis of existing frameworks and their dimensions leads to a number of observations:

[3] Recently IAF has included the "Enterprise" main architecture area, which comprises one holistic representation of the organization as a whole.

- They use different terms for similar aspects, and similar terms for different aspects (for example: the term "business" in IFW is not the same as the term "business" in TOGAF).
- They often define terms only informally making it difficult to demarcate boundaries clearly (for example: where does the conceptual level end and the logical level begin?).
- They often do not name dimensions explicitly, leaving their interpretation up to the reader (an example is the March framework).
- They sometimes do not distinguish clear values within the dimensions, hindering effective communication (an example is the Evernden Eight that leaves the exact content of all dimensions up to the reader).
- They often have slightly different sets of values for particular dimensions (see for example the IAF "design phases" dimension and the Zachman "perspectives" dimension).
- They sometimes have dimensions with values that do not have a clear relationship, which makes it hard to understand the dimension altogether (take for example the "special viewpoints" dimension in IAF).

These observations show that architecture frameworks are not the silver bullet for the confusion that exists when talking about architecture. Not only do individual frameworks leave us with some questions, but current architecture frameworks are also inconsistent with each other, making it necessary to tell someone which framework you use when talking about architecture. What is required in our view is an underlying concept of architectural dimension, but more on that later.

6.2.3.2 Essentials

Analysis of the frameworks also leads to another important observation, in that frameworks in essence are an attempt of the creator to enable clustering of architectural information in a way that suits a particular context and goal, with many parties involved. We see dimensions that are unnamed, which may express the lack of one overall concept. We perceive these unnamed dimensions as a struggle by the creator to capture in one stroke the main dimensions by which the various stakeholders structure their world.

We observe a division of dimensions into primary, secondary and supporting ones. The reason probably is that two dimensions are often enough to cover the required architectural descriptions. Also, on paper it seems most natural to represent the framework as a collection of cells, spread over the two dimensions of a matrix; one dimension is depicted horizontally (primary dimension), and another one vertically (secondary dimension). Sometimes other (supporting) dimensions are shown, or mentioned in the description of the framework.

The primary, horizontal dimension is often the type of information (topic), which can be divided into business and IT aspects. This distinction between business and IT is prominent in enterprise-class frameworks, but missing in application-class frameworks. This is perfectly explainable from the purpose of the framework: enterprise-class frameworks need to align business and IT, while application-class frameworks only need to model an IT solution. A general observation concerning this first dimension is that, although relationships exist, the values can be described fairly independent of one another.

The secondary, vertical dimension, in contrast, often is one that has a sequential aspect or is simply a partitioning in different levels of detail. With a sequential aspect there is a certain order in the construction of architectures that follows the values in this dimension. Examples of such dimensions are the IFW dimension “levels of constraint”, and the IAF dimension “design phases”. When devising such a dimension the framework creator must discover which architectural descriptions need to be fixed first, and which architectural descriptions need to be based on them. When the dimension is a partitioning into levels of detail, the higher rows contain a higher level of abstraction (fewer details) than lower levels. These two meanings of the secondary dimension (sequence versus levels of detail) are very similar since a design usually progresses from a high level of abstraction to a lower level of abstraction.

Dimensions are inherent in the paradigms people use, and prevailing paradigms can be a good source for concepts to build architectural dimensions from. Examples of these are the chain of control, the value chain, and the phases in development.

These observations might be helpful for those that want to describe their own architecture framework. They need to ask themselves: “what is the purpose of this framework?”, “what are the types of information that need to be described?” and “what is the order in which we want to architect?”. We believe that the best architecture framework is the one that provides answers that are most appropriate for a specific context.

6.3 Dimensions

We have used the term “dimension” informally several times in this paper already. It is an everyday word. Now we will try to formalize it. Using the resulting definition we will synthesize a list of base dimensions from existing frameworks. These base dimensions are further explained and illustrated subsequently.

Webster Online offers the following explanation of the word “dimension”:
1 a (1) **:** measure in one direction; *specifically* **:** one of three coordinates determining a position in space or four coordinates determining a position in space and time (2) **:** one of a group of properties whose number is necessary and sufficient to determine uniquely each element of a system of usually mathematical entities (as an aggregate of points in real or abstract space) <the surface of a sphere has two *dimensions*>; *also* **:** a parameter or coordinate variable assigned to such a property <the three *dimensions* of momentum> (3) **:** the number of elements in a basis of a vector space **b :** the quality of spatial extension **: MAGNITUDE, SIZE c :** a lifelike or realistic quality **d :** the range over which or the degree to which something extends **: SCOPE** -- usually used in plural **e :** one of the elements or factors making up a complete personality or entity **: ASPECT**
2 *obsolete* **:** bodily form or proportions
3 : any of the fundamental units (as of mass, length, or time) on which a derived unit is based; *also* **:** the power of such a unit
4 : wood or stone cut to pieces of specified size
5 : a level of existence or consciousness

With a little play of words from **1 a** (2) we like to see a dimension in the field of IT-architecture as an attribute of a piece of information which positions this piece of information in the total available information space. **1 e** shows that more than one dimension is needed to make up a complete architecture description. And **3** speaks of fundamental (base) units on which a derived (practical applicable) unit is based. So far the (serious) play of words. If we try to put it in one sentence, it would be something like:

An architectural dimension is a criterion to partition an architectural description into a set of segments, where each segment is identified by a unique value within a list of values associated with the dimension.

Architectural descriptions should document the dimensions used and the segments they cover in an introductory chapter. Standardizing these dimensions, their segments in particular, in a specific organizational context prevents semantic obscurities and introduces a shared architecture terminology.

6.3.1 Base dimensions

Based on our definition and existing architecture frameworks, we will now synthesize a list of nine base dimensions. The sources of inspiration for these dimensions are the existing architecture frameworks. We have studied the dimensions in these frameworks and transformed them into "pure" dimensions conforming to our definition. The resulting list is shown in Table 6-2, which includes a short description of the dimension and a hint at possible values. Since it is not our intent to standardize or formalize these values, they are just meant to illustrate the dimension. Also, we are not claiming that the set of nine dimensions is complete; other base dimensions may exist and could be added to our list.

Dimension	**Description**
Type of information	The topic of the information *(business, organisation, technical)*
Scope	The extent of the information covered *(industry sector, organisation, domain, system family, system, component)*
Detail level	The amount of detail *(high, medium, low)*
Stakeholder	The target audience *(client, end-user, architect, analyst, developer)*
Transformation	The transformation phases that the architecture needs to cover *(current situation, short-term, medium-term, long-term)*
Quality attribute	The quality attribute that is being addressed *(functionality, reliability, usability, efficiency, maintainability, portability)*
Meta level	The amount of abstraction *(instance, model, meta-model, meta-meta-model, meta-meta-meta-model)*
Nature	The nature of the information *(policy, principle, guideline, description or standard)*
Representation	The way architectural information is represented *(formal, semi-formal, informal)*

Table 6-2 Proposed base dimensions

6.3.2 Base dimensions in detail

We will now describe our proposed base dimensions in more detail. The first five are fairly common in architecture frameworks. The other four are used less frequently.

Type of information – This dimension is by far the most prevalent in architecture frameworks, and describes the subject of architectural information. Another way to look at this dimension is that it consists of the concepts that exist in domain-specific languages. At a high level this dimension can distinguish segments such as business, organisation, and technical. Within these segments a further segmentation typically exists. For example, IFW decomposes the technical segment into interface, network and platform segments. Some other frameworks that use this dimension are: 4+1, DYA, GEM, GRAAL, RM-ODP, Siemens, and TOGAF.

We perceive this dimension as a means to break down a complex situation into more or less independent aspects. Together these aspects provide a conceptual model of the entire environment. Some frameworks are explicit about the relationship between aspects. An example is the GRAAL framework which claims a service provisioning sequence from “processing and networking hardware” to ”software platform” to “software applications”, and so further. TOGAF prescribes a design sequence from “business” to “information systems” to “technology”, which we read as a claim that the business determines the information systems, and that information systems determine the technology.

Scope – This dimension describes the scope of the information covered. It is our proposal for a “clean” top down dimension, one that is easily understood. One way to decompose this dimension is with the values industry, organisation, organisational domain, system family, system, and system component. Scope is the main dimension of the Gartner framework [Rosser02] with a different list of values. Different interpretations of the dimension are possible, interpretations that each may be valid from a specific point of view. The scope dimension is very much related to the ownership dimension in IFW, and it is implicitly used in the levels of constraint dimension in IFW. In particular, the design context and operational bound values in IFW have a system scope, while the upper levels have a domain scope.

Detail level – This dimension is based on the amount of detail, where levels with more information can be defined. A characteristic is that all information of the level above is kept, and that new information is added. The primary goal of varying the level of detail is to leave out those details that are not relevant or known in a particular context or at a particular moment in time. Since it is possible to add different types of detail, one could say that the detail level dimension comes in various types. Examples of frameworks in which we recognize a detail level dimension are: Zachman (perspectives dimension), IAF (design phases dimension), March (second unnamed dimension). We say “recognize” because the detail level portioning is a bit obscured by other meanings attached to these dimensions in the frameworks.

Stakeholder – This dimension uses the stakeholders that are addressed as primary criterion. Stakeholders are typically only interested in certain parts of the architecture. Defining descriptions for specific stakeholders was the intention of the Zachman perspectives dimension, but this also holds for other architecture frameworks such as 4+1, IAF and RM-ODP. Again, the pure meaning of the “stakeholder” dimension is obscured by other meanings attached to it in the frameworks.

Transformation – The transformation dimension uses change in time as the criterion. It distinguishes the current situation from short-term, medium-term and long-term situations, including the transitions between them. A slightly different way to define this dimension is to not refer to specific moments in time, but to characteristics of the situation that can exist in time, like the levels in the Capability Maturity Model (CMM) initial, repeatable, defined, managed and optimised. Examples of frameworks that use this dimension are: Gartner, IFW, Evernden Eight and March.

Quality attribute – A number of dimensions in existing frameworks mention quality characteristics such as security, performance and usability; see for example IAF and Maier/Rechtin. In our view these characteristics can be considered as a separate dimension, with segments that each highlights certain quality characteristics. The values within this dimension are defined by quality frameworks. Various quality frameworks exist, such as the Extended ISO model [Zeist96]. This dimension makes it possible to talk about, for example, a performance view or a security view. These last two views are also very common types of quality-driven views.

Meta level – This dimension addresses those architectures that, instead of domain-specific models, provide general classifications and relationships. It really describes a meta-model; information about information. Consider for example a model that describes the types of components that may be developed, and the legal relationships between them. Multiple meta levels exist (meta-meta models, and so forth), but arguing that these are architectural in nature becomes increasingly difficult. The "meta level" dimension resembles the "detail level" dimension; the difference is that instead of less information meta-models describe different information. Evernden Eight is a framework that includes a meta level dimension.

Nature – This dimension determines the nature of the architectural information; is it a policy, principle, guideline, model or standard. Inherent in this dimension is the extent in which designers need to comply with the architectural information. A policy is clearly more important to follow than a guideline. The dimension is based on the dimensions as defined by Boar and DYA.

Representation – This dimension uses the way to represent architectural information as criterion. One can choose between formal, semi-formal and informal representations. An informal representation is natural language, which leaves room for interpretation. Semi-formal means such as UML improve the well-definedness. Formal description languages such as C2 and Rapide [Medvidovic00] are at the other extreme, but sometimes necessary to automatically generate models or reason about them. For example, a performance model based on Queueing Networks [Smith02] provides a very accurate description of a system. The Evernden Eight framework also includes a representation dimension.

6.3.3 Usage

The list of base dimensions can be used in many different ways: as communication vehicle, checklist or basis for an architecture description or an architecture framework. The primary goal of the list is to facilitate communication about architecture in general. There are several ways to support this, such as documenting the values that an architectural description covers in the various dimensions in an introductory chapter. Also, in verbal communication these dimensions can be used to position an architectural description. Using the list as checklist allows one to check whether all relevant aspects have been taken into account for a specific architecture. Finally, the list can be used in the construction of a new architectural description or architecture framework. This means selection of the most applicable dimensions and values within those dimensions, and translating those to document structures.

6.3.4 Example

We will now exemplify the use of our list of base dimensions by positioning the view on architecture of the Rational Unified Process (RUP) [Rational02], an object-oriented software development method (see Table 6-3). The software architecture document (SAD), as RUP calls the architectural description, contains seven potential viewpoints that are inspired by the 4+1 model. In addition to the original viewpoints, also a data and user experience viewpoint are added. Looking at these viewpoints we see that they describe technical information about the system. RUP talks about the "software architecture" of a system, implying a system scope for the architecture. The detail level of the information is medium; it is not the intention of the SAD to be a detailed design. Looking at the activities that the SAD is input to, we derive that the target audience of the document are the designers and implementers of the system. The goal of the SAD is to be a short-term architecture; project members need to fully comply to it immediately. Although the emphasis of the SAD is on the functionality of the system, the impact on all other quality attributes also needs to be documented. The contents of the document are models of the system; no meta-models are described. Also, the nature of the architectural information is that it contains only models; no principles, guidelines or standards. The models are represented by UML models, which are supplemented with text.

Base Dimension	Value
Type of information	Technical
Scope	System
Detail level	Medium
Stakeholder	Designer, Implementer
Transformation	Short-term
Quality attribute	All
Meta level	Model
Nature	Model
Representation	Text, UML diagram

Table 6-3 Positioning architecture within Rational Unified Process

6.4 IEEE 1471

The IEEE 1471 "recommended practice" defines concepts and their relationships that are relevant for architectural descriptions [IEEE1471]. It also provides guidance on the structure of architectural descriptions. The main concepts standardised are "architecture", "architectural description", "concern", "stakeholder", "viewpoint" and "view", see Figure 6-3. 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". Architectural

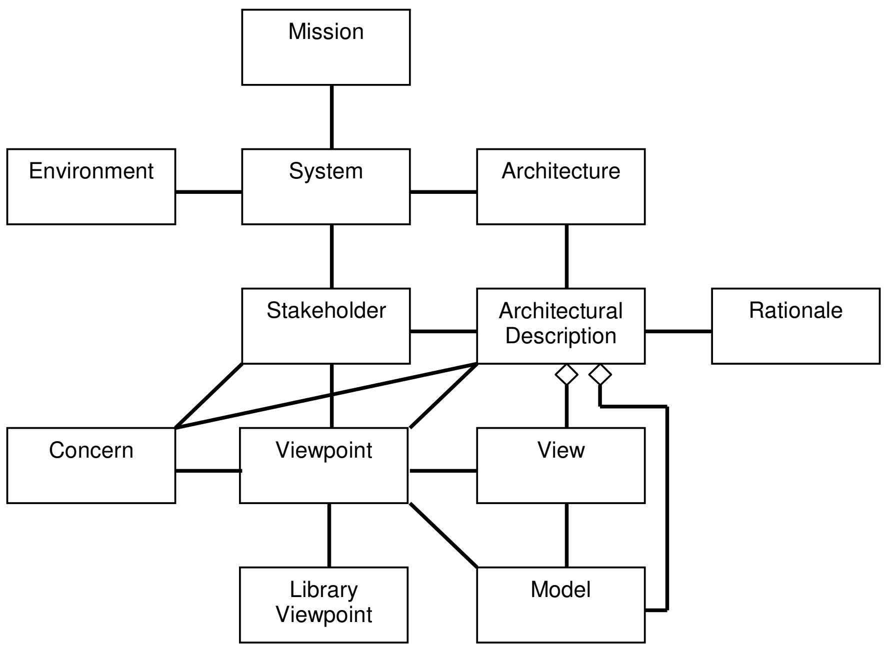


Figure 6-3 IEEE/ANSI Std 1471 conceptual model

descriptions are segmented into views. A view addresses a related set of stakeholder concerns and is constructed in accordance with the specification that is laid down in a viewpoint. Together the views address all the concerns of the stakeholders.

Since its publication in 2000, IEEE 1471 has received much appraisal. The concepts of stakeholders, concerns and views are accepted as essential. The terminology proposed by IEEE 1471 is now being used by many architects. The focus on concerns of stakeholders is a good stimulus for otherwise possibly too technically oriented IT architects. After all, it is the interests of the stakeholders that need to be served.

Although IEEE 1471 is an important contribution to standardising architecture terminology, it still leaves a number of things unspecified. Most importantly, IEEE 1471 does not propose nor prescribe any specific viewpoint, which might confuse architects and stakeholders. In a specific context, two architects can easily disagree on who the stakeholders and their concerns are, and what information is needed to address these concerns. Also, if a view contains a lot of architectural information, it needs to be structured, bringing back the needs for which frameworks have been defined. This also holds at the enterprise level, where many IEEE 1471 compliant architectural descriptions may need to be made accessible. So, even if all IT architects would follow IEEE 1471, architectural information could still be very different up to a point where documents are still not accessible, nor comparable. The “dimension” concept provides a means to further structure IEEE 1471 views into more manageable chunks.

We don’t write this in critique of IEEE 1471, but we do feel compelled to raise some arguments against a view of IEEE 1471 as the silver bullet where it comes to architectural descriptions. Also with IEEE 1471 at hand, there still is a need for additional support to help communication about IT architecture.

We also see a mismatch between IEEE 1471 and existing architecture practice as represented by the frameworks in our overview. IEEE 1471 requires a view to address a set of related concerns. The “chunks” in which existing frameworks divide the architectural information are addressing many concerns, but it is not obvious these concerns are “related” in the sense of IEEE 1471. Our guess is they aren’t, but a difficulty here is that IEEE 1471 does not specify what “related” exactly means.

6.5 Conclusions and future work

There are many differences between existing architecture frameworks. Partly this can be explained from their original goal, and the context from which they originated. A commonality is that architectural information is often organised in a matrix that is bound by two dimensions: one dimension typically addresses the type of information, and a second one having a sequential order.

In this paper, we propose the use of nine base dimensions: Type of information, Scope, Detail level, Stakeholder, Transformation, Quality attribute, Meta level, Nature and Representation. These base dimensions allow us to better understand and compare existing frameworks, or to create a new framework. They also ease the understanding and communication of architectural descriptions.

There still remains a lot of work to be done in architecture description standardization. In particular, the values within the dimensions described need to be widely agreed upon. This will lead to standardized architectural viewpoints (library viewpoints), and will ultimately contribute to the further maturation of the architect profession. We would like to understand more of the circumstances in which the different frameworks function.

We recommend the use of IEEE 1471 and would like to see more constructive debate to come to effective application of this standard.

6.6 Acknowledgements

We want to thank everyone that has provided us with helpful comments and feedback. In particular, we thank Philippe Spaas, Harry Hendrickx, Jan Campschroer, Erik Vermeulen, and Michiel Perdeck for their review comments. We also want to thank the anonymous reviewers of an earlier version of this paper.

6.7 References

[Boar98] Bernard H. Boar: “Constructing Blueprints for Enterprise IT Architecture”, Wiley, 1998.

[Baat99] J.M. de Baat: “CMG's Multi-Channel Management Vision on Architecture”, October 1999.

[ESPRIT93] “CIMOSA - Open System Architecture for CIM”, ESPRIT Consortium AMICE, Springer-Verlag, Berlin, 1993, (ISBN 3-540-56256-7), (ISBN 0-387-56256-7).

[Evernden96] R. Evernden: “The Information FrameWork”, IBM Systems Journal, 1996.

[Evernden02] R. Evernden: “Evernden Eight”, 4th resource, 2002.

[Goedvolk99] H. Goedvolk, D. Rijsenbrij: “White Paper Integrated Architecture Framework”, version 1.0, 1999.

[Hermans02] L. Hermans: “Uitbuiten synergie ICT- en business strategie”, Informatie, ten Hagen Stam, September 2002.

[Herzum00] P. Herzum, O. Sims: “Business Component Factory”, John Wiley & Sons, 2000.

[Hofmeister95] C. Hofmeister, R.L. Nord, D. Soni: “An Industrial Perspective of Software Architecture”, In Proceedings of the Eleventh International Conference on Data Engineering, IEEE Computer Society, Taipei, Taiwan, March 1995.

[Hofmeister00] C. Hofmeister, R. Nord, D. Soni: “Applied Software Architecture”, Addison Wesley, 2000.

[IEEE1471] IEEE Std 1471-2000: “IEEE Recommended Practice for Architectural Description of Software-Intensive Systems”, 2000.

[IFIPIFAC98] “GERAM: Generalised Enterprise Reference Architecture and Methodology”, IFIP–IFAC Task Force, Version 1.6.2, June 1998.

[ISO10746] ISO/IEC CD 10746-1, “Basic Reference Model of Open Distributed Processing”, 1994.

[Kruchten95] P. Kruchten: “The 4+1 View Model of Architecture”, IEEE Software, 1995.

[Lassing01] N. Lassing, D. Rijsenbrij, H. van Vliet: “Viewpoints on modifiability”, International Journal of Software Engineering and Knowledge Engineering, Vol. 11, No. 4, 2001, 453-478.

[Maier02] M.W. Maier, E. Rechtin: “The art of systems architecting”, CRC Press, 2002.

[Medvidovic00] Nenad Medvidovic, Richard N. Taylor: “A Classification and Comparison Framework for Software Architecture Description Languages”, IEEE Transactions on Software Engineering 26, 1 (2000), pp 70-93.

[Meinema99] J.L. Meinema: "Corporate architecture: a conceptual approach", University of Twente, 1999.

[OpenGroup03] The Open Group: "The Open Group Architectural Framework", Version 8.1.

[Rational02] "Rational Unified Process", Rational Corporation, Version 2002.05.20.005, 2002.

[Rosser02] B. Rosser: "Defining Architecture for IT: A Framework of Frameworks", Gartner, Research Note, 12 August, 2002.

[Scheer92] A.W. Scheer : "Architecture of Integrated Information Systems", Springer, 1992.

[Smith02] C.Smith, L. Williams: "Performance Solutions", Addison Wesley, 2002.

[Sowa92] J.F. Sowa and J.A. Zachman: "Extending and formalizing the framework for information systems architecture", IBM Systems Journal 31, No. 3, 1992.

[Tapscott93] D. Tapscott , D. Caston: "Paradigm Shift - The New Promise of Information Technology", McGraw-Hill, 1993.

[VanEck02] P.A.T. van Eck, H. Blanken, M. Fokkinga, P.W.G. Grefen, R.J. Wieringa: "A Conceptual Framework for Architecture Alignment Guidelines - Project GRAAL WP1 Whitepaper", October 17, 2002, Department of Computer Science, University of Twente.

[Wagter01] R. Wagter et al.: "DYA: snelheid en samenhang in business en ICT-Architecture", Tutein Nolthenius, 2001.

[Youngs99] R. Youngs et al.: "A standard for architecture description", IBM Systems Journal, Vol. 38, No. 1, 1999.

[Zachman87] J. A. Zachman: "A Framework for Information Systems Architecture", IBM Systems Journal, Vol. 26, No. 3, 1987.

[Zeist96] B. van Zeist et.al.: "Kwaliteit van software producten, praktijkervaring met een kwaliteitsmodel", Kluwer Bedrijfsinformatie, 1996.

7 A Lightweight Method for the Modelling of Enterprise Architectures – Introduction and Usage Feedback

Henk Koning, Rik Bos, Sjaak Brinkkemper*
Department of Information and Computing Sciences, University of Utrecht, P.O.Box 80.089, 3508 TB Utrecht, the Netherlands, {h.koning, rik, s.brinkkemper}@cs.uu.nl

Abstract: This paper introduces a technique developed at the computer science department of Utrecht University to facilitate enterprise architecture design and modelling. It defines the high level key concepts necessary to describe enterprise architectures in a lightweight manner that is easy to create and to understand. For this purpose we introduce the Enterprise Architecture Modelling method (EAM), driven by our teaching requirements and based on our experience in ERP modelling and on a reflection on managerial decision taking. EAM consists of these types of diagrams: the Supply Chain Diagram, showing the relationships of the enterprise in its business environment; the Enterprise Function Diagrams for the interoperation of enterprise functions; the Scenario Overlay for modelling the main business processes; the System Infrastructure Diagram, depicting the technical infrastructure of IT systems and networks; and the Application Overlay Diagram, showing which applications give support to which enterprise functions. To date, we conducted about 40 case studies in different sectors of society. Diagrams from one of the case studies illustrate the usage of EAM in this paper. To solicit feedback we performed an enquiry among modellers and readers of EAM reports. We now offer this technique to the wider EA community and welcome further feedback.

Keywords*:* enterprise architecture, modelling method, views, lightweight, feedback, management decisions

7.1 Enterprise architecture

Within large organizations in the public and private sector, the subject of Enterprise Architecture (EA) becomes ever more important to support strategic decision making. EA descriptions capture the essentials of the business and link them to the essentials of the supporting information technology (IT) infrastructure. EA models need to be clearly understandable by business managers as well as by IT-specialists. Therefore the language in text and diagrams should be intuitively clear, without a need to learn many technical terms or icons. Despite its importance, we are still far away from a universal architecture language, let alone a universal approach for constructing enterprise architectures. Most existing approaches are burdened under lots of details, while we believe that the main focus of EA should be on models that capture only the essentials, that are yet easy to communicate, and from which it is therefore also relatively easy to take strategic decisions upon. At the Utrecht University we have been successfully experimenting for a few years with more focused and yet easy models. We have come to a stage in which we want to share these models with the wider community.

7.1.1 Motivation

Nowadays many complex issues reside on the agenda of the board of the organization (EWITA 2008), (Ward and Peppard 2002). Issues like:

Organization. Should we change the division of enterprise functions because of new products and services? How can we accommodate the consequences of the merger?
Outsourcing. Should we outsource certain enterprise functions? Which? How?
IT support. Should we start implementing an ERP system in that unit? When can we replace system X with a standard product? For which enterprise functions do we introduce new information systems?
Service continuity. Is our IT infrastructure trustworthy for 7x24? Do we get back to our customers in time?

These are some issues and many more could be given. It is not possible for one modelling method to tackle all issues that exist somewhere in the IT domain. The focus of EAM is on issues surrounding the relation between the main building blocks of the organisation (enterprise functions) and the main building blocks of the IT-support (information systems). To be able to evaluate the consequences of strategic IT-related decisions the board needs an *insightful overview* of the business and of the existing and prospective IT-support. The board is usually not interested in detailed information that is needed for developing information systems or for guidance in the daily operations. Based on our experiences and on a review of existing enterprise modelling approaches (ArchiMate (Lankhorst *et al.* 2005), DEM (Van Es and Post 1996), SEAM (Wegmann 2003) and ARIS (Scheer 1998a, 1998b)), we say there are four key areas of interest to look at for this insight and overview:

- *Enterprise context*. Who are the customers and how are they serviced? Which other parties are involved in our business? What products, services or information are exchanged? What is the chain of dependencies between us and the other parties?
- *Enterprise functions*. What are the main business functions of the company? How do they interoperate to produce the goods and services rendered? With which external parties do they deal? What information is exchanged? What are the sub functions? How does this company work?

- *Information systems*. Which information systems support the enterprise functions? Where is support lacking? Or redundant? When can a tailor-made system be replaced by a commercial application? Do the information systems fit the business?
- *IT infrastructure.* What processing servers (computers) and networks are available to run the information systems and give access to them? Does the IT-infrastructure fit the business?

We have geared our method to support these key areas of interest. In our method we stick to a high level of abstraction that is in general sufficient (and desired) for treating subjects at the managerial level. For specific issues that may arise once in a while, supplementary information can be compiled by the IT-department covering those issues. We deliberately take into account the computers and network equipment, because they absorb a considerable amount of money and need timely planning to be able to follow adjustments in the information systems or in the business operations. For the organization structure we rely on documentation that is already available in each company. In our view a 'pure' decomposition of *enterprise functions* offers in the long run the most stable anchor point for the design of IT-solutions.

The four key areas of interest can be seen as *perspectives* that make up an enterprise architecture description. We have chosen to work with different partial perspectives instead of one overall perspective. Partial views enable better understanding and verification in the aspects treated in a particular partial view (Parsons 2002). The underlying meta model, see section 7.2.2, guarantees the consistent integration of the partial views. It would need further research to assess the necessity and sufficiency of the four perspectives.

The board has the need to decide and communicate in an efficient and transparent manner. Enterprise architectures are one of the means to assist these types of decisions. Therefore the enterprise architect is confronted with the question of what documentation should be provided and what communication should be performed to support transparent decision making? The efficiency of decision making is also heavily dependent on the simplicity of the formalism (Hargis 2000). So the research question for this paper is:

How can enterprise architectures be described in a way that is easy to create (author) and easy to understand (reader)?

With *easy* creating and understanding we mean: the proper meaning can be established quickly, use of the right level of abstraction (it gives fitting answers to questions that live), easily checked for correctness, no information lacking or redundant, etc. See section 7.1.2.

The research question requires that we first clarify our understanding of the concepts of enterprise and enterprise architecture. An *enterprise* is defined as a legal entity that provides products or services to other legal entities or consumers. Enterprises can be companies in the private sector, or business units of a company, but also organizations in the public sector, so even associations or foundations. We define *enterprise architecture* as the collection of texts and models concerning the complete information infrastructure of an enterprise in relation to the enterprise functions. With *information infrastructure* we mean the information technology components that support the enterprise functions, like information systems, computers and network equipment. Lankhorst *et.al.* (2005, p. 3) give this definition of Enterprise Architecture (EA) "a coherent whole of principles, methods and models that are used in the design and realisation of an enterprise's organisational structure, business processes, information systems and infrastructure". We further limit the scope of EA to "*decision taking concerning* design and realisation of …".

Based on earlier experiences with enterprise modelling for ERP implementations we have developed an Enterprise Architecture Modelling method (EAM). The objective of EAM is to capture the essentials of an enterprise with respect to the architecture and at the same time facilitating easy communication for managerial decision making. This method was inspired by various high level business modelling formalisms introduced by vendors of ERP software; ARIS (Scheer 1998a, 1998b) for SAP, and Dynamic Enterprise Modelling (DEM) (Van Es and Post 1996) for Baan implementations. EAM contains no revolutionary new modelling techniques, but is, on the contrary, assembled from existing techniques using method engineering (Brinkkemper *et al.* 1999). SCD, EFD en SO are mainly derived from DEM which contains similar diagrams, AO is inspired by landscape maps, see (Torre *et al.* 2005). SID is a very old type of diagrams, see for instance (Bell *et al.* 1976). The novelty is the combination of techniques, the abstraction level, the strict tailoring to the information needs of enterprise architecture, and the ease of use.

For two years EAM has been instructed to master students of Business Informatics in an elective course. The students get the assignment to perform case studies using EAM within real life companies. Furthermore, some master graduation projects were executed using EAM as modelling formalism.

As it turned out, the results obtained by the students were very satisfactory given the restricted scope and time. We were pleased by the clear understanding of the essential operating of the companies that showed from the student papers. Also the contact persons from the organizations were very satisfied with the resulting papers. We will present parts of one of these case studies in order to introduce EAM in section 7.3.

7.1.2 Rationale for a lightweight method

In the design and development of EAM many factors have played a role. We wanted to keep EAM simple: not too many concepts, not too many details and a simple presentation. In our experience this is what managers want. We feel backed in this by scientific research. Miller (1956) has shown that the human brain is only capable of processing around seven pieces of information at the same time. So that is a strong driver to keep it simple. According to Chan (2001) the managerial decision quality deteriorates under information overload. Again a driver to be concise.

Eppler and Mengis (2003, p. 21) state that “Regarding information itself, the essential mechanisms to fight information overload are to assure that it is of high value, that it is delivered in the most convenient way and format, that it is visualized, compressed and aggregated, and to use signals and testimonials to minimize the risks associated with information“. EAM is apt at presenting the information visualized and compressed. The expertise of the enterprise architect is adding value to his output. Eppler (2004) has studied the communication problems between experts and managers, and has found, amongst others (page 15): “Summarizing these issues, we can conclude that experts struggle with three major issues when transferring their knowledge to managers: First, reducing or synthesizing their insights adequately, second, adapting these trimmed insights to the management context without distorting them, and third, presenting the compressed and adapted findings in a trust-building style and reacting adequately to management questions and feedback." EAM provides means in the synthesizing/compressing and adapting to management context, as will be explained in the sections 7.2 and 7.3.

Koniger and Janowitz (1995) have looked at ways of identifying quickly the relevancy of information. They claim “A continuous refinement of the information selection process is necessary in order to cope with the rapid growth of produced data.“ (page 10). EAM aims at being selective in offering just the right type of information.

Finally, Simpson and Prusak (1995) argue that the value of information for managers is not in the amount of information, but in the attributes truth, guidance, accessibility, scarcity and weight. These attributes are highly determined by the professionalism of the enterprise architect. The manager expects guidance regarding decisions to be taken. Using simple models enhances the accessibility.

7.1.3 Related work

Several methods for developing enterprise architectures are available nowadays, e.g. ArchiMate (Lankhorst *et al.* 2005), DEM (Van Es and Post 1996), SEAM (Wegmann 2003) and ARIS (Scheer 1998a, 1998b). Each method has its own strengths and usually has its focus on one or more specific points, e.g. integration between different models, alignment between business and IT, business processes, communication etc. We mention some methods here briefly to compare them to our goals of easy creating and reading, and of giving a high level overview.

The ArchiMate project has produced an elaborate language for describing enterprise architectures. The conceptual model consists of 29 entities with a corresponding visual representation consisting of 40 symbols each having a specific meaning. It takes a while to get to learn each of the symbols and, in our experience, after some time of not working with the language one has to relearn the specific meaning of each of the symbols. The authors of ArchiMate have proposed 16 diagram types (viewpoints) as a basic set to work with the language, but many more could be constructed. Archimate has, for our goal unneeded, concepts to model the organisation at a detailed level, to model interfaces between the levels, to model the internal structure of applications and all the services that are provided by the infrastructure. The ArchiMate diagram type 'Business Function Viewpoint' comes very close to our Enterprise Function Diagram. They state 'Business functions are used to represent what is most stable about a company in terms of the primary activities it performs, regardless of organisational changes or technological developments' (p. 177).

ARIS originated as a method for describing business processes in the context of SAP implementation processes, but has developed into a thorough, general purpose EA modelling tool. It is even more complex than ArchiMate. It has 5 basic views, but numerous diagram types to populate the views. On the summary page (Scheer 1998a, p. 78) we count 23 'main' diagram types. The business process meta-model contains 300 entities and relations (p. 48). For students the time needed to master the basics of ARIS is not proportional to the analysis time needed to study the enterprise architecture of a company. Apart from giving a high level overview, ARIS is also meant to support the detailed design and evaluation of business processes and to support the design of information systems. The emphasis in ARIS on business processes leads to a too detailed view of the organization to serve the enterprise architectural purposes.

The well known framework of Zachman (Sowa and Zachman 1992, Zachman 1987) has 36 different viewpoints to give aid in categorizing the architectural information, but gives no guidance regarding the modelling of the information. The same goes for an architecture process description like TOGAF (Open group 2002). It gives guidance regarding the activities of the architect, but does not have a modelling method. The 4+1 framework of Kruchten (1995) gives an architectural structure and modelling method, but is more geared toward software architecture and not applicable for enterprise architecture.

Dynamic Enterprise Modelling (DEM) (Van Es and Post 1996) was introduced by Baan in 1996 as a means for implementing the Baan ERP product. The modelling focused on a Petri-net based technique for business process modelling to which the Baan application units were to be linked. DEM also contains a supply chain diagram tool for the logistic network of the company and of an enterprise function modelling diagram. The latter two diagrams formed an inspiration for our EAM method. The Petri-net based business process modelling of DEM turns out to be too detailed for modelling enterprise architectures.

The team of Wegmann of the École Polytechnique Fédérale de Lausanne has developed an object-oriented enterprise architecture method, called “Systemic Enterprise Architecture Methodology” (SEAM) (Wegmann 2003). As part of SEAM, they have developed the CAD tool “SeamCAD” (Le and Wegmann 2008). SeamCAD enables the modelling of hierarchical systems (spanning from business down to IT) at different levels of detail (e.g. from large business transaction to detailed interactions). It has a philosophical underpinning which is not so easy to understand and doesn’t give guidance as to what modelling concepts should be used or what levels should be distinguished. The notation is UML-like and some training is needed to read the diagrams.

MEMO, Multi-perspective Enterprise Modelling, was developed by Frank (2002). He proposes a framework of three so called perspectives - strategy, organization and information system - each of which is structured by four aspects: structure, process, resources and goals. MEMO contains three modelling languages: strategy modelling language (MEMO-SML), organization modelling language (MEMO-OrgML) and object oriented modelling language (MEMO-OML), which allow for detailed modelling of the three perspectives. The EAM method we propose later on in this paper, seems to be a subsubset of MEMO’s strategy goals, organization structure and process, and information system resources. MEMO has an interesting setup, but it falls short of our wishes regarding ‘easy to create’ and ‘easy to understand’.

Braun and Winter (2005) describe an Enterprise Architecture meta model which has four layers: strategy, organization, application, software. The models of the first three layers are shown in a slightly simplified manner, and the relationships between these models are elaborated. A successful implementation of these models, using a meta modelling tool, is reported. The three simplified models contain respectively 27, 27 and 22 concepts. So, in our view, they are not easy to learn, they entice a lot of detailing and underline the need we feel to start anew with a basic method containing few concepts. We like the limited number of layers, with the focus on linking the organization to the IT-support.

UML, the Unified Modelling Language (Object Management Group 2003b), needs mentioning because it is nowadays the most well known modelling language, and serves as a reference point for many people. We chose not to use UML, because UML has its strength in object oriented modelling of the internals of information systems and that is not the abstraction level of enterprise architecture. The premier modelling concept in UML is the ‘class’, with operations and data attributes, a concept we don’t use in describing enterprise architectures as it is too detailed to serve at the enterprise level. In general UML is too much fine grained for what we need in enterprise architecture.

7.1.4 Outline of the paper

In the next section we present an overview of the EAM method with the key concepts and a short outline of the steps in the modelling process. In section 7.3 we present each of the models of EAM and illustrate them with example diagrams from the case study. In section 7.4 we outline our efforts to receive usage feedback by a large series of case studies and two questionnaires on the ease of creation and on understandability. We finish the paper with conclusions and future work.

7.2 Enterprise architecture modelling

In our view EA should describe the essential functioning of an enterprise and the essential functioning of the IT-support for the enterprise. These essentials lay the groundwork which shapes the more detailed functioning of the organization and of the IT; therefore it is called architecture. Our EAM method endeavours to give a high level modelling of an enterprise and its IT-support. The primary purpose is to give support in strategic decision taking regarding IT. The EAM models serve as a frame of reference for subsequent system analysis and design activities. We call EAM a lightweight method. By this we mean it has a simple conceptual model, it is easy to use and the resulting artefacts are easy to understand.

In this section we will first present the diagramming structure of EAM, then its main concepts by means of a meta model, and then some guidelines for the process of modelling enterprise architectures.

In the diagrams, colours can be used to denote categories of enterprise functions, e.g. one colour for all financial functions, see (Koning *et al.* 2002) for this use of colour. In the meta model colour is an attribute of Enterprise Function.

7.2.1 Diagramming tools for enterprise architectures

The EAM method consists of the following diagrams:

A *Supply Chain Diagram* (SCD) shows how the enterprise works together with business partners to produce the goods or services for the customers (enterprise context).

An *Enterprise Function Diagram* (EFD) gives a top level breakdown of the main functions of an enterprise. The top diagram covering the complete enterprise is called the *corporate EFD*, and the lower level EFDs are called *function EFD*.

A *Scenario Overlay* (SO) shows how the enterprise functions in an EFD interoperate in a particular situation.

An *Application Overlay Diagram* (AO) shows which applications give support to which enterprise functions.

A *System Infrastructure Diagram* (SID) shows the main network topology, the main computers that function in the network and the main information systems that run on these computers to support the enterprise functions.

In Fig 1 the overview structure of these models is shown.

We will explain these models each in turn in section 7.3. With each we give an example diagram taken from a case study performed at the company Center Parcs.

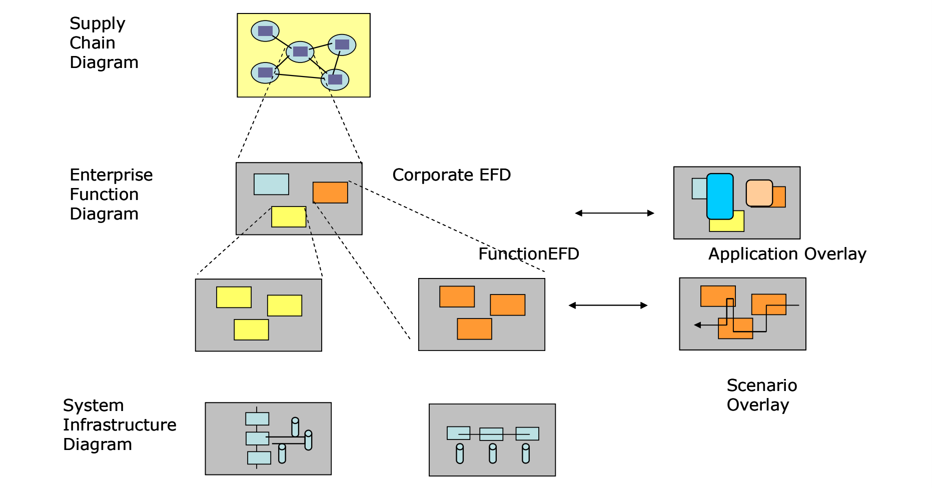


Fig.1. Overview of EAM models

7.2.2 Concepts of Enterprise Architectures

We have studied existing EA modelling methods and made a choice of key concepts needed for capturing high level essentials. See Fig 2 for the meta-model of our modelling method. Note that this somewhat simplified model does not contain all constraints as it only shows our key concepts and how they are related. [4]

For enterprise we use The Open Group (2002) definition “Any collection of organizations that has a common set of goals and/or a single bottom line. In that sense, an enterprise can be a government agency, a whole corporation, a division of a corporation, a single department, or a chain of geographically distant organizations linked together by common ownership”. For the sake of simplicity, when dealing with external parties, we include in this definition any collection of *individuals* (e.g. customers) that have a common set of goals and/or a single bottom line (and for which the enterprise to be modelled develops policies and services).

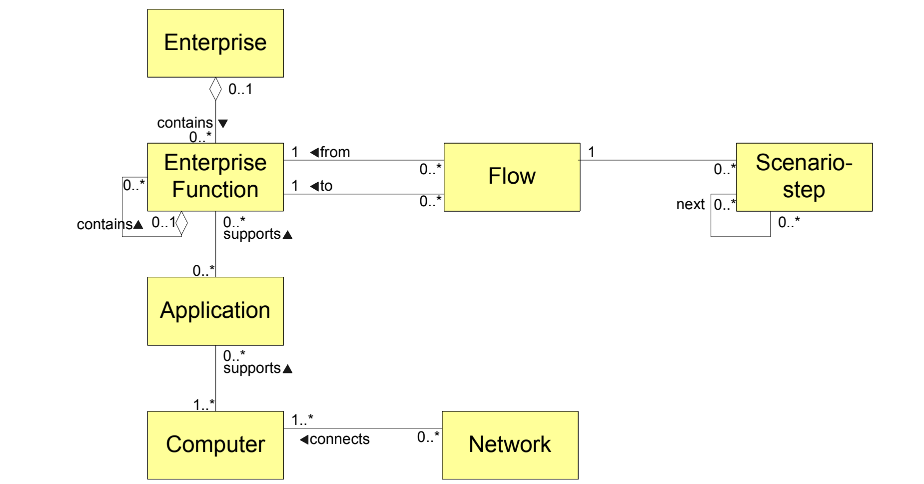


Fig.2. The meta-model of the Enterprise Architecture Modeling method (EAM)

In modelling the essentials we take into account the four key areas of interest as named in the introduction: enterprise context, enterprise functions, information systems and infrastructure. To model the context of an enterprise we use enterprises connected by flows between their enterprise functions. For modelling the enterprise the key concept is the enterprise function. We define an enterprise function as a collection of coherent processes, continuously performed within an enterprise and supporting its mission. To show the interoperation of the enterprise functions we also portray the flow (of information or products & services). Scenarios indicate a sequence of flows. For modelling the information systems and the infrastructure our key concepts are computer, application and network (component). We use ‘computer’ as a general term to indicate all sorts of processing units or executional components. Likewise ‘network’ stands for all sorts of connectivity components. We have indications that these key concepts are sufficient for creating a model that shows how in essence an enterprise operates, see section 7.4. With these the current IT-support can be evaluated by the business management at a high level of abstraction and future IT-support can be planned. In section 7.3 we will describe the models and indicate with each on what meta-model concepts they are based.

7.2.3 Modelling process

In Fig 3 we have depicted a schematical representation of the modelling process of EAM. In our experience the best way is to create the Supply Chain Diagram in conjunction with the Enterprise Function Diagram at the corporate level. Then proceed with the EFDs at the functional level. When these are elaborated, the main processes can be portrayed in

[4] Additional textual constraints are for example ‘an enterprise function cannot send a flow to itself’ and ‘an enterprise function cannot contain itself’.

Scenario Overlays; this is also a check on the EFDs. The AO shows the application support for the enterprise functions. Finally, the System Infrastructure Diagram is produced.

Even though Fig 3 suggests a more or less linear path, in practice a lot of iterations need to be preformed, for instance, caused by feedback from validating the models, or caused by changing requirements for enterprise functions or infrastructure.

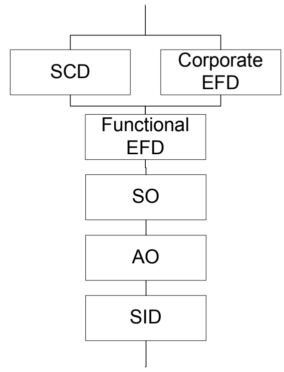


Fig. 3. Modeling process

7.3 The EAM models

In this section we present each of the models of EAM and illustrate them with example diagrams from a case study at Center Parcs Europe. For each diagram type we have tried to keep the diagrammatic language as simple as possible to achieve maximum clarity.

Center Parcs Europe (CPE) is one of Europe's largest companies in the accommodations rental for short holidays. Its headquarters is in Rotterdam, the Netherlands and over 10,000 full-time and part-time employees are currently working at CPE all over Europe. Currently, CPE offers about 10,000 bungalows and cottages in 20 parks. Four departments in every park take care of the customers. These departments are housekeeping, catering, retail and leisure. In 2003, CPE welcomed over 3 million guests while the turnover was about 525 million Euros.

7.3.1 Supply Chain Diagram (SCD)

The Supply Chain Diagram is a model of the *enterprise context* of the enterprise together with its business partners and the exchange of products and services. Supply Chain Diagrams create a quick overview of the enterprise as a whole and of the main players in its enterprise contexts. The SCD is based on the meta-model concepts 'enterprise', 'enterprise function' and 'flow'. In reality the flows connect enterprise functions in the different enterprises, but in the graphical presentation of the SCD these underlying enterprise functions, are suppressed, see (van Buuren *et al.* 2004) for this technique. An SCD shows the business relationships of the enterprise in a kind of dependency network. The enterprises in an SCD are not decomposed further, but for large companies business units can be treated as separate enterprises. An EA description contains one SCD.

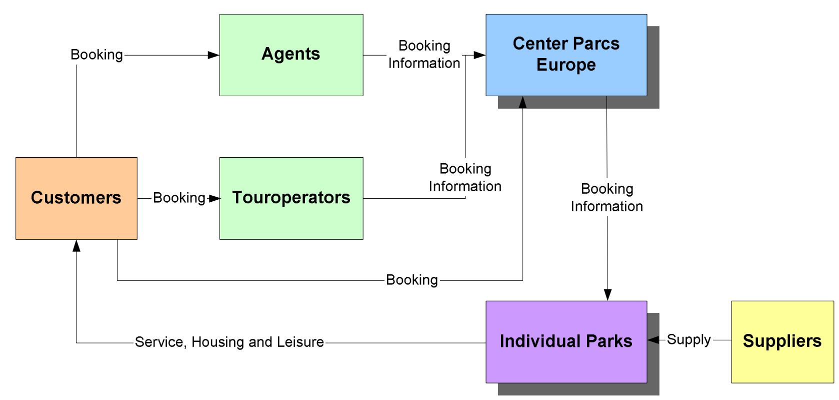


Fig. 4. SCD of Center Parcs

See Fig 4 for an example of an SCD. It shows how CPE and the Individual Parks cooperate with Agents, Tour operators and Suppliers to accommodate Customers. The boxes denote enterprises, the arrows the flow of products and services, or of information.

In this example the two main units of CPE are shown, the central Europe headquarters and all the individual parks (in one box), and the main external parties which we describe here one by one.

Customer The customers (visitors) of CPE are able to make a booking in three different ways: (1) directly at CPE via the website or the call center, (2) via a tour operator, who will subsequently handle all the customer contacts, or (3) via an agent, who will pass through the customer information to CPE.

Tour Operator and Agent When a customer makes a booking via a tour operator or agent, the latter is responsible for passing the booking information to CPE. Tour operators are also responsible for the financial handling while CPE takes care of the financial handling for agent bookings.

Suppliers The suppliers primarily deal with the individual parks. They receive orders directly from the departments in the parks and ship their goods directly to the parks.

7.3.2 Enterprise Function Diagram (EFD)

An Enterprise Function Diagram is a model from an *enterprise function* perspective. EFDs give a top level breakdown of the main operations of an enterprise with their

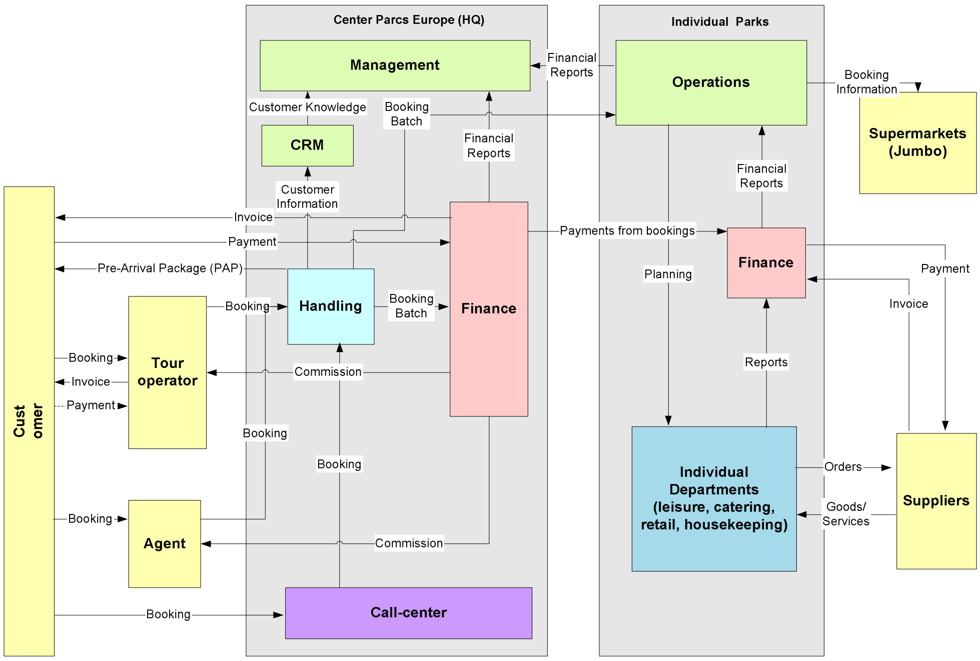


Fig. 5. EFD with the main enterprise functions of Center Parcs

information flows.

See Fig 5 for an example of an EFD. The boxes denote enterprise functions. To the left and to the right the external parties are portrayed. Arrows indicate the flow of information. The EFD is based on the meta-model concepts ‘enterprise’, ‘enterprise function’ and ‘flow’; this is equal to the SCD but now the focus is on the enterprise functions. To give you an idea of the meaning of this diagram we discuss each of the functions briefly.

7.3.2.1 Center Parcs Europe (HQ) – Enterprise Functions

Call center Handles bookings from customers, made by either telephone or via the website.

Handling Deals with all incoming bookings from call center, tour operators, and agents.

Finance Receives information about the bookings that are made from the Handling

function, sends invoices to the customers, and receives the payments from them (not if the customer booked via a tour operator). Finance will also calculate and pay the commission to the tour operators and agents. The payments from the bookings are transferred to the individual parks (which are all separate corporations). Finance also reports to the Management function.
Management Receives and reviews financial reports from the Finance functions. Planning and feedback on the financial reports are sent to the Finance functions.
CRM The Customer Relationship Management function is responsible for all CRM-related activities. It aims to transform customer information into knowledge (e.g. customer profiles and segments, etc.).

7.3.2.2 Individual parks – Enterprise Functions

Individual Departments Are not further elaborated, since they all perform the same functions from the perspective of EA. They receive a planning from the Operations function, and report to the Finance function. They are all individually responsible for the ordering of supply.
Finance Receives payments from the CPE HQ, handles invoices from and payments to the suppliers. Reports to the Operations function.
Operations Responsible for all the planning. They receive reports from the Finance function, and have to report to the management of the CPE HQ.

7.3.2.3 Functional EFD

Enterprise functions in an EFD can be decomposed in the same diagram or in a separate EFD. See Fig 6 for an example of a decomposition. A tree structure of EFDs can be set up to analyze the architecture of an enterprise. Case study evidence (see section 7.4) shows that usually two levels are enough to get sufficient grip on the complexity of an organization. We call the top level EFD a *corporate* EFD (see Fig 5.) and a decomposition a *functional* EFD.

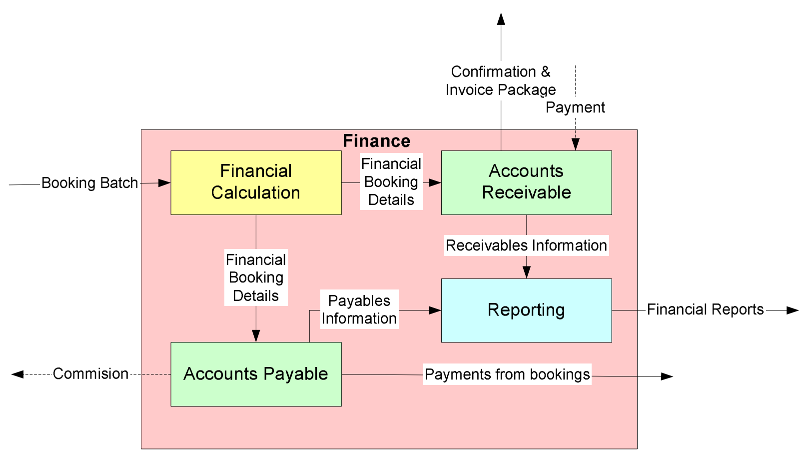


Fig. 6. Decomposition of the Finance function of Center Parcs Europe

The EFD of Finance shows the following sub functions.
Accounts Receivable Responsible for all incoming payments. For this, it receives booking details from the Financial Calculation function. It sends out booking confirmations and invoices to the customers and collects payments from the customers. Accounts Receivable reports to the Reporting function.
Accounts Payable Responsible for all outgoing payments, mainly commissions to the tour operators and agents, and payments collected from the customers. Details of the payments

that are to be paid are determined by the Financial Calculation function. Accounts Payable reports to the Reporting function.
Financial Calculation Responsible for all computations regarding bookings and commissions. It sends this to the Accounts Receivable and Accounts Payable.
Reporting Receives financial information and prepares financial reports that are sent out to the Management function.

EFDs should be distinguished from the more general Data Flow Diagram (Yourdon, 1989). EFDs have the emphasis on enterprise functions and their mutual data exchange and not on the detailed processing of data as in a DFD. Furthermore EFDs are restricted to a high level of abstraction.

7.3.3 Scenario overlay (SO)

A scenario is a continuous processing of a request trigger by various enterprise functions, which results in one or more feedback triggers. A Scenario Overlay provides insight in the interoperation of *enterprise functions* and in the completeness of the EFD. A scenario is drawn as an extra diagram level on top of an EFD with a proper explanation. Only essential flows are elaborated into a scenario (highest frequency, large impact). The scenario overlay adds limited, but for our goal sufficient, process information to the EFD. It gives fewer details than process models that have been created with a (dedicated) process modelling language. See Fig 7 for an example. Red lines are drawn that touch EFD functions in the execution of the process that is triggered. The extra information for the scenario, compared to the underlying EFD, is based on the meta-model concept 'scenario step'.

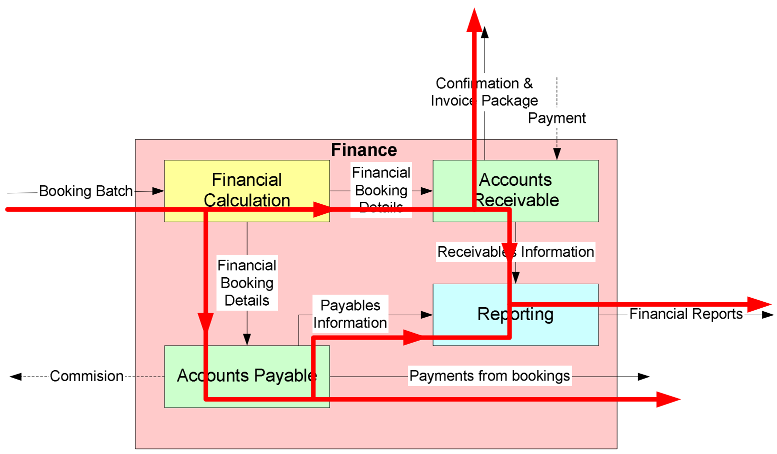


Fig. 7. Example Scenario Overlay, the financial processing of a Booking Batch

Here we show one scenario overlay for the Finance function. This scenario concerns a booking made via the CPE call center, so there is no commission. The Financial Calculation function determines the exact price of the booking. This information is sent to both the Accounts Receivable and the Accounts Payable functions. The Accounts Receivable sends a confirmation and invoice package to the customer. It sends all information regarding unpaid and paid amounts to the Reporting function. Accounts Payable will divert the payment from the customer for his booking to the respective park, and forwards information about the payments to Reporting. The Reporting function within the Finance function prepares financial reports, which are sent to the Management function.

Scenarios are an optional part of an EA description. Typically an EA description will contain several SOs.

7.3.4 System infrastructure diagram (SID)

A SID shows the *information systems* and information technology *infrastructure* of an enterprise, or a well-defined part thereof. A SID shows the main network topology, the main computers that function in the network and the main information systems (applications) that run on these computers to support the enterprise functions. The SID is based on the meta-model concepts ‘application’, ‘computer’ and ‘network’. A SID contains two types of computers. There are executional computers (or systems): Applications, Workstations, Servers (data, web, file, applications), Databases, Fire-walls. And there are connectivity computers (or Network): Internal Network (LAN, WAN), External Network (private: Leased lines, public: Internet).

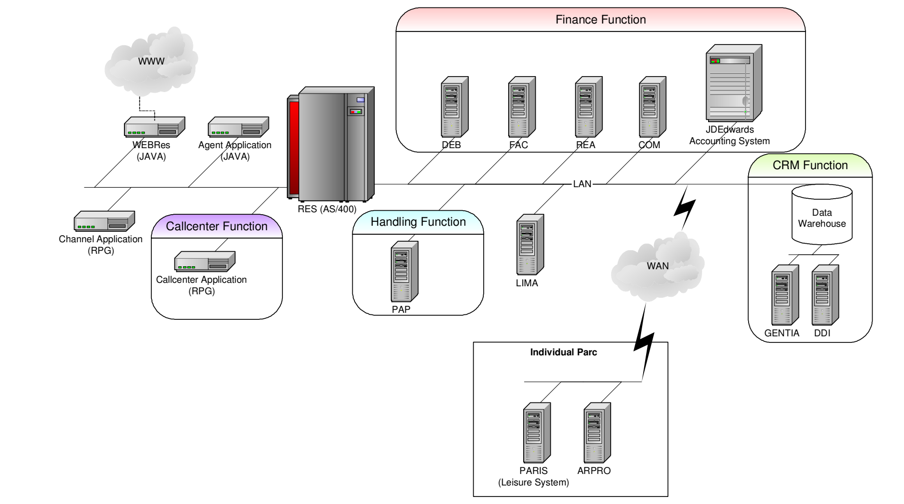


Fig. 8. Example System Infrastructure Diagram, a mixture of platforms at CPE

For SIDs there are many popular notational variants. An EA description typically contains one SID, corresponding to the abstraction level of the corporate EFD. Complex parts in this SID can be exposed by creating a SID at a more detailed level.

See Fig 8 for an example. The computer icons depict hardware systems, the adjoining acronyms stand for software systems (applications) that run on them. The hardware systems are grouped by the enterprise functions they support. The lines depict network connections. The central computer depicted in the system infrastructure diagram is the Reservation (RES) system, running on an IBM AS/400 mainframe. On the left side of the RES system, the computers and applications that handle the incoming bookings are shown. Of the computers on the right side of the RES system we mention only briefly the main functions and we do this per enterprise function they belong to:
Handling function The preparation and mailing of the pre-arrival packages.
Finance function Accounts receivable, invoicing, financial reporting, calculation and payment of the commission for the agents and tour operators, Ledger system.
Individual Park Bookings of extra activities in the parks, keep track of information about arriving customers.
CRM function Customer history, central data storage, statistical analysis and data mining system.

As can be seen in the infrastructure diagram, CPE uses many different systems.

7.3.5 Application overlay (AO)

To fill a gap that is felt between the Enterprise Function Diagram and the System Infrastructure Diagram we later introduced a new type of diagram, comparable to the Scenario Overlay, the Application Overlay (AO). The applications are drawn as an extra diagram level on top of an EFD with a proper explanation. The AO deals with information systems (applications) in their own right. Information systems (software components) have a different lifecycle than computers and network (hardware components). The AO is based on the meta-model concepts 'enterprise function' and 'application'. See Fig 9 for an example. This simple example was not part of the Centre Parcs case study, but was constructed by the authors of this paper as an example. This AO shows that the Bookkeeping application supports the enterprise functions Accounts Payable and Accounts Receivable, Calculation supports Financial Calculation, and Reporting Tool supports Reporting. Note, that we use generic application names, instead of the real commercial names for the sake of neutrality.

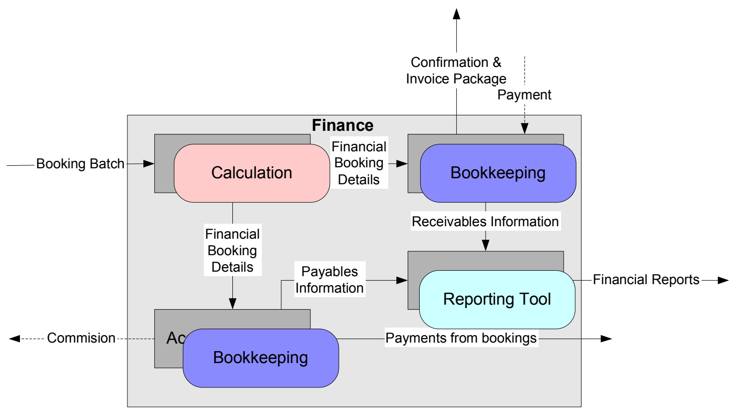


Fig. 9. Example Application Overlay , applications that support the (greyed out) Finance enterprise functions of Center Parcs Europe

The AO shows clearly which enterprise functions are supported, once or multiple times (possibly a sign of redundancy of fragmentation), and which functions are not supported by IT. If an application supports enterprise functions that are not adjacent in the underlying EFD diagram, then the application will be drawn more than once (see Bookkeeping in Fig 9).

7.4 Usage feedback

We felt a necessity to solicit feedback from the usage of EAM in the enterprise architecture practice. We have conducted two simple surveys to get some indications regarding our goals of easy to create and easy to read. In the current stage of introducing and developing EAM, we feel we can involve students as subjects in our study, see for instance (Boehm *et al.* 1998). Tichy (2000) claimed that especially in case of exploratory research students can play the role of young professionals. We have not arrived at a stage yet were we can test EAM with IT-professionals as creators, but we have started to promote EAM among IT-professionals, so maybe it is possible in the future.

7.4.1 Course Enterprise Architecture

Each fall a master course on Enterprise Architecture (see (Utrecht University 2005) for a general description) is given as an elective course in the international MSc program of Business Informatics at Utrecht University. An important assignment in this course is to do a case study in practice, meaning that the students, who already have a bachelor in computer science, have to produce a complete EA model for a real company, which includes all diagram types. A representative of the company had to consent to the models, and declare they really represented the given situation at the company. In the first year we had 40 students in 19 groups working at 15 different companies. In the second year we had 68 students in 23 groups working at 18 different companies. Six companies participated for the second time, making up a total of 27 different organizations. The case studies were performed by students as part of their academic training. They were not performed to evaluate EAM, but are very valuable as practical experience with EAM.

In Table 1 we list the distribution of the companies among different categories in the public and private sector according to a standard categorization of companies of the Netherlands Bureau of Statistics (1993). Except for the three categories AFH (Agriculture, Forestry and Hunting), Mining and Education, all categories are represented by at least one company. So this is a pretty good coverage.

Table 1. *Organizations per category*

Category	Organizations	#
Public Utilities (water, electricity, gas)	Gasunie, Rendo	2
Construction Industry	NCL	1
Transportation, Communication, Shipping	KLM, Schiphol, NorfolkLine	3
Finance	RABO Bank, AAV, HDN	3
Services	Accenture/UnIT, AAC Cosmos, BiZZdesign, Conclusion, Exact, GTI, Morgan Chambers, VNO-NCW	8
Public Administration	Dutch Tax Admin., Social Security Admin., Municipalities (2)	4
Health Care	Academic Medical Center	1
Industry	Cisco, Thales, Sanoma	3
Tourism, Recreation and Sport	Center Parcs, NOC*NSF	2

The students are placed in groups of 2 or 3 each and before they visit the companies for the interviews, they get lectures on our EAM technique together with some small exercises. Lectures and exercises take about eight hours.

Even though practical modelling is an important and substantial part of our course, we provide overviews of other important topics, like frameworks or approaches, e.g. Zachman’s framework (Sowa and Zachman 1992, Zachman 1987), ARIS (Scheer 1998a, 1998b), TOGAF (Open group 2002), IEEE Std 1471 (IEEE, 2000), ‘4+1’ View Model (Kruchten 1995), Enterprise Unified Process (Ambler 2005), SOA (Krafzig *et al.* 2005), MDA (Object Management Group 2003a),.

In the case studies we tried to investigate how well the models could be communicated. At an informal level most people involved (both technical and non-technical) agreed that the models were easy to understand and that the models also gave a quick impression of the essential functions within a company. Moreover, it turned out that certain characteristics within the diagrams and (in)consistencies between the diagrams could easily be traced and clarified.

Furthermore, EAM was used in practice in several projects in the industry. Here we mention the following since these are published. First, at a municipality to decide on a new e-government portal service to be launched (Zuiderhoek *et al.* 2006). Second, the integration of enterprise applications at the Royal Netherlands Army was performed by applying EAM. EAM was extended with some UML diagrams to design the integrated business processes spanning multiple enterprise applications, see (Roques *et al.* 2007).

7.4.2 Questionnaire

We launched two surveys to solicit usage feedback of EAM: one for the ease of creation and one for the usability.

7.4.2.1 Ease of creation

We launched a questionnaire among the students who performed the case studies. See Table 2 for a summary of the results. Six questions related to the ease of creation were asked for each of the diagram types in EAM. The Application Overlay Diagram was not taken into account in this questionnaire, as it was introduced later. The students answered the questions immediately after performing the case study, and with respect to their own work. We feel this was a viable thing to do, because we have a very open relationship with our students and in generally they don't spare us their criticism. They are graduate students, many of them will start working as consultants after their study, some have their own company, while others already worked for IT companies. So many of them are quite mature and very capable of giving their opinion.

We have no comparable figures concerning other methods and the students had no normative references for their answers. So we don't take the outcome of the survey as a vindication in an absolute sense. We look at relative differences in the figures or at extreme values (or the voluntary remarks that were added to the answers) to point us to aspects of EAM that need attention:

Table 2. Results of the ease of creation questionnaire among students (n=23)

Question	Supply Chain Diagram	Enterprise Function Diagram	Scenario Overlay	System Infra-structure Diagram
Is the .. diagram easily readable? Rate on a scale from 1 (very bad readable) to 5 (very good readable) how good the .. diagram readable is.	4,3	3,2	3,7	3,8
Has the .. diagram the right level of abstraction? Make a choice: a. less detail preferred / b. just right the way it is / c. more detail preferred.	a 4% b 74% c 22%	a 9% b 91% c 0%	a 9% b 86% c 5%	a 14% b 54% c 32%
Is the correctness of the .. diagram easily established (conformity with the reality within the company)? Rate on a scale from 1 (very difficult) to 5 (very easy) how well the .. diagram can be checked.	3,5	2,7	3,5	3,4
Is there information lacking on the .. diagram? Chose y (yes, information is lacking) or n (no, no information is lacking).	y 35% n 65%	y 35% n 65%	y 23% n 77%	y 24% n 76%
Is there redundant information in the .. diagram? Chose y (yes, there is redundant information) or n (no, there is no redundant information)	y 4% n 96%	y 9% n 91%	y 5% n 95%	y 10% n 90%
How easy is it to produce this kind of diagram on the basis of available information? Rate on a scale from 1 (very difficult) to 5 (very easy) how well the .. diagram can be produced.	3,7	2,7	3,6	3,3

- the readability of the EFD diagram, (remark: too many arrows)
- the abstraction level of SCD and SID (remarks: SCD more parties, business units, only core business, SID (no remarks))
- the correctness of the EFD, (remarks: a lot of experts needed; a lot of outsourced processes)
- information lacking on all diagram types (remarks: SCD money flows? Data flows? High abstraction level; EFD only most important functions modelled, no room for company structures, maybe missed something; SO loops, timeframes, departments, maybe missed something, omitted some flows, conditions; SID maybe missed something)

- makeability of EFD (remarks: more guidelines, not all companies think in functions, need more information, what to do with exceptions).

7.4.2.2 Understandability

Table 3. Results of the Understandability questionnaire among readers (n=11)

Question	Supply Chain Diagram	Enterprise Function Diagram	Scenario Overlay	System Infra-structure Diagram
Is the .. diagram easily readable? Rate on a scale from 1 (very bad readable) to 5 (very good readable) how good the .. diagram readable is.	4,5	3,7	3,2	3,4
Has the .. diagram the right level of abstraction? Make a choice: a. less detail preferred / b. just right the way it is / c. more detail preferred.	a 0% b 45% c 55%	a 0% b 91% c 9%	a 0% b 64% c 36%	a 0% b 67% c 33%
Is the correctness of the .. diagram easily established (conformity with the reality within the company)? Rate on a scale from 1 (very difficult) to 5 (very easy) how well the .. diagram can be checked.	3,9	3,7	3,3	3,9
Is there information lacking on the .. diagram? Chose y (yes, information is lacking) or n (no, no information is lacking).	y 82% n 18%	y 40% n 60%	y 64% n 36%	y 50% n 50%
Is there redundant information in the .. diagram? Chose y (yes, there is redundant information) or n (no, there is no redundant information)	y 9% n 91%	y 18% n 82%	y 0% n 100%	y 0% n 100%
How well usable are this kind of diagrams in your organization? Rate on a scale from 1 (totally unusable) to 5 (very well usable) how usable the diagram is.	3,7	3,7	3,2	3,7

To complement this internal evaluation we also put out the questionnaire among our contact persons at the participating companies, see Table 3. The sixth question (how easy to produce) was replaced by a ‘consumer’ question (how well to use). These contact persons mainly had an IT background, and were themselves responsible to report to management on IT issues.

When we look for relative differences with the student score, these are the points of interest that surface:

- the level of abstraction and the lacking of information of the SCD, these indicate that the readers would like more details on the SCD, they want a diagram with more information on it. The students found the SCD the easiest to make. So here is an attention point.
- higher scores for EFD, the trouble students seem to have with the EFD is not repeated with the readers (probably because they are familiar with the business domain that is described in the EFD).
- lower scores for SO, the readers appreciate the SO less than the students

- the lacking of information on all diagram types, not only on SCD but on all types the readers want more information

7.5 Conclusions and future work

In this paper we have presented the Enterprise Architecture Modelling (EAM) method consisting of five diagram types for modelling enterprise architectures at a high level in a fast and simple way. 40 Different case studies were performed using EAM. We conducted two small scale questionnaires. We conclude that EAM for the moment meets our objectives of 'easy to create' and 'easy to read'. For authors EAM can be learned and trained in a short course of one day. The resulting diagrams can be understood without any specific training.

Further research is needed to assess the necessity and sufficiency of the four perspectives for managerial decision making. It is required to incorporate non-IT staff in future evaluations of EAM. We want to continue developing EAM and take into account the attention points coming out of questionnaires. For the authors the creation of the EFD needs attention, for the readers the lack of information needs to be sorted out, especially on the SCD.

Besides these points we want to produce tool support for EAM, and, together with a partner from industry, we want to develop the practical application of EAM.

What started as a mere conviction has now been tried out on a modest scale and usage feedback has been received. We would like to see EAM used on a larger scale in the EA community, which will give us hopefully more feedback on the strengths and weaknesses of EAM.

7.6 Acknowledgements

We thank Ronald Bos and Inge van de Weerd for conducting the case study at CPE.

7.7 References

Ambler, S.W., Nalbone, J. & Vizdos, M., 2005. The Enterprise Unified Process – Extending the Rational Unified Process. Prentice Hall PTR.

Bell, T. E., Bixler, D. C., and Dyer, M. E., 1976. An extendable approach to computer-aided software requirements engineering. *In*: *Proceedings of the 2nd international Conference on Software Engineering* (San Francisco, California, United States, October 13 - 15, 1976). International Conference on Software Engineering. IEEE Computer Society Press, Los Alamitos, CA, 70.

Boehm, B., Egyed, A., Kwan, J., Port, D., Shah, A., and Madachy, R., 1998. "Using the WinWin Spiral Model: A Case Study", *IEEE Computer*, July 1998, pp. 33-44

Brinkkemper, S., M. Saeki, and F. Harmsen, 1999. Meta-Modelling Based Assembly Techniques for Situational Method Engineering. *Information Systems*, vol. 24, No. 3, pp. 209-228.

Braun, C., Winter, R., 2005. A Comprehensive Enterprise Architecture Metamodel and Its Implementation Using a Metamodelling Platform. *In:* Desel, J., Frank, U. (Eds.): *Enterprise Modelling and Information Systems Architectures*, Proc. of the Workshop in Klagenfurt, GI-Edition Lecture Notes (LNI), Klagenfurt, 24.10.2005, Gesellschaft für Informatik, Bonn, P-75, 2005, pp. 64-79.

Buuren, R. van, H. Jonkers, M.-E. Iacob & P. Strating, 2004. Composition of relations in enterprise architecture models. *In*: H. Ehrig et al. (eds.), *Graph Transformations: Second International Conference* (LNCS 3256), Rome, Italy, Sept. 2004, pp. 39-53.

Chan, S.Y., 2001. The use of graphs as decision aids in relation to information overload and managerial decision quality, *Journal of Information Science*, Vol. 27, No. 6, 417-425.

Eppler, M.J., 2004. Knowledge Communication Problems between Experts and Managers. An Analysis of Knowledge Transfer in Decision Processes, ICA Working Paper #1/2004, University of Lugano, Lugano

Eppler, M.J. & Mengis, J., 2003. A Framework for Information Overload Research in Organizations. Insights from Organization Science, Accounting, Marketing, MIS, and Related Disciplines, #1/2003 ICA Working Paper, University of Lugano, Lugano.

Van Es, R.M., Post, H.A., 1996. Dynamic Enterprise Modelling, Kluwer.

EWITA, 2008. *Enterprise Architecture* [online]. Avialable from: http://www.ewita.com/EA_Overview/EADefinition.htm [Accessed July 5, 2008].

Frank, U., 2002. Multi-perspective Enterprise Modelling (MEMO) - Conceptual Framework and Modelling Languages, *Proceedings of the annual Hawaii international conference on system sciences*.

Hargis, G., 2000. Readability and computer documentation, ACM Journal of Computer Documentation, August 2000, Vol.24, No. 3.

IEEE, 2000. IEEE recommended practice for architecture description, IEEE Std 1471.

Koniger, P. & Janowitz, K., 1995. Drowning in Information, But Thirsty for Knowledge. *International Journal of Information management*, 15 (1), 5-16.

Koning, H., Dormann, C., van Vliet, H., 2002. Practical Guidelines for the Readability of IT-architecture Diagrams. *In*: *Proceedings ACM SIGDOC 2002*, pp 90-99.

Krafzig, D., Banke, K. & Slama, D. 2005. Enterprise SOA – Service-Oriented Architecture Best Practices. Prentice Hall PTR.

Kruchten, P., 1995. Architectural blueprints – The '4+1' View Model of Software Architecture, *IEEE Software*, 12(6); 42 – 50.

Lankhorst, M. et al, 2005. Enterprise Architecture at Work, Springer-Verlag Berlin.

Le, L.S., Wegmann, A, 2008. SeamCAD 1.x: User's Guide [online]. Available from: http://infoscience.epfl.ch/getfile.py?mode=best&recid=55774 [Accessed July 5, 2008].

Miller, G.A., 1956. The Magical Number Seven, Plus or Minus Two: Some Limits on Our Capacity for Processing Information, *The Psychological Review*, vol. 63, pp. 81-97

Netherlands Bureay of Statistics (Centraal Bureau voor de Statistiek): *Standaard Bedrijfsindeling 1993 (SBI '93)* [online]. Available from: http://www.cbs.nl/nl-NL/menu/methoden/classificaties/overzicht/sbi/default.htm [Accessed July 5, 2008].

Object Management Group, 2003a. *MDA Guide version 1.0.1*, Miller, J. & Mukerji, J (eds.) [online]. Available from: http://www.omg.org/docs/omg/03-06-01.pdf [Accessed July 5, 2008].

Object Management Group, 2003b. *UML 2.0 Superstructure Final Adopted specification*, Document reference ptc/04-10-02 August 2003. Available from the Object Management Group website: www.omg.org.

The Open Group, 2002. *TOGAF "Enterprise Edition" Version 8.1* [online]. Available from: http://www.opengroup.org/architecture/togaf8-doc/arch/ [Accessed July 5, 2008].

Parsons, J., 2002. Effects of Local Versus Global Schema Diagrams on Verification and Communication in Conceptual Data Modelling. *Journal of Management Information Systems*, 19(3): p. 155-183.

Roques, J., Vader, H., Bos, R. & Brinkkemper, S., 2007. SAIM - A situational method for application integration. UU-CS (Int. Rep. 2007-022). UU WINFI Informatica en Informatiekunde.

Scheer, A.W., 1998a. ARIS – Business Process Frameworks, Springer-Verlag Berlin.

Scheer, A.W., 1998b. ARIS – Business Process Modelling, Springer-Verlag Berlin

Simpson, W. & Prusak, L., 1995. Troubles with information overload - moving from quantity to quality in information provision. *International Journal of Information Management*, 15(6), 413-425.

Sowa, J.F. & Zachman, J.A., 1992. Extending and formalizing the framework for information systems architecture. *IBM Systems Journal*, vol 31(3).

Tichy, W. F., 2000. 'Hints for Reviewing Empirical Work in Software Engineering', *Empirical Software Engineering*, 5(4), pp. 309-312.

Torre, L.W.N. van der, Lankhorst, M.M., Doest, H. ter, Campschroer, J., Arbab, F., 2005. Landscape maps for enterprise architectures, report SEN-E0514, National Research Institute for Mathematics and Computer Science, The Netherlands

Utrecht University, 2005. *Course Enterprise Architecture* [online]. Available from: http://www.cs.uu.nl/education/vak.php?vak=INFOEIA&jaar=2007 [Accessed July 5, 2008].

Wegmann, A, 2003. ON THE SYSTEMIC ENTERPRISE ARCHITECTURE METHODOLOGY (SEAM), *proceedings ICEIS 2003*.

Ward, J. & Peppard, J., 2002. Strategic Planning for Information Systems. Third Edition. Wiley Series in Information Systems. John Wiley & Sons Ltd.: West Sussex, England.

Yourdon, E., 1989 Modern structured analysis, Yourdon Press, Upper Saddle River, NJ,

Zachman, J.A., 1987. A framework for information systems architecture. *IBM Systems Journal* 26(3); 276 – 292.

Zuiderhoek, B., Otter, A., Bos, R. & Brinkkemper S., 2006. Framework for Dutch Municipalities to Ensure Business IT Alignment Using Enterprise Architecture. *Proceedings of the 6th European Conference on e- Government*, pp.457-466, Academic Conferences International, Reading, England

8 Conclusions of this research

This chapter offers a summary of the answers to our research questions, we give a recap of our findings regarding IEEE Std 1471, we list the contributions of this research and we outline our ideas for future research.

8.1 Research questions and answers

8.1.1 Q1 How can the readability of IT-architecture diagrams be improved?

To answer this question we have compiled a list of guidelines. See chapter 2 and appendix A. Our list of guidelines concerning visual attributes has the following sections: hierarchy/layers, forms/size/width, layout, color, connectors, text, and graphics & icons.

We have used the guidelines twice in commercial workshops with around 10 attendants. The goal was to give professionals an approach to evaluate and improve their own diagrams. After the introduction of the guidelines it was very interesting to see how the participants knew how to improve their diagrams. Positioning was straightened out, superfluous information was omitted and colors were added or improved. Publication of the guidelines on the website of this research resulted in emails of appreciation every now and then.

We mention here the principal solutions we found, which are worked out more concretely in the guidelines. In crowded diagrams a clear hierarchy of two or three levels should be created. Sizes and widths should be equal for similar objects. A clear pattern and horizontal and vertical alignment speed up the processing of the diagram. Natural positions should be preferred. Color is a very strong visual attribute with some striking peculiarities. It should be used, but with restraint. Light, non primary colors are safe to use. There are solutions in dealing with color blindness or black and white printing . There are a few modest guidelines to slightly alleviate the connector mess. Well-chosen text can give strong figurative support and stimulate thinking. Meaningful icons and graphics can enhance the visual appeal and ease of use of a diagram.

As part of our research activity we have presented these guidelines to three groups of aproximately 10 IT-architects from the companies that supported our research. After three months a questionnaire was filled in by these persons to indicate their support or not for each of the guidelines. On the basis of this questionnaire 32 guidelines were removed from the list of in total 190 draft guidelines.

8.1.2 Q2 How large is the gap between current architecture documents and IEEE Std 1471?

The main trait of IEEE Std 1471 is the organization of an architecture description in concern related views. For four investigated non-IEEE Std 1471 architecture descriptions we have established the relation of the content to stakeholder concerns, see Table 3-1 to Table 3-4. We call these ‘relevancy patterns’. The tables show that stakeholders are mostly interested in only part of the document, often less than half of it and that parts of the answers to their concerns are scattered all over the document. This makes it difficult to devise a uniform scheme for breaking up the documents into smaller, concern related parts. Except for one document there is no clear dividing line for breaking up the documents.

We conclude that for three of the four investigated documents the gap in relation to IEEE Std 1471 is large. The answers to the stakeholder concerns are scattered all over the document and it is very hard to rearrange the content to divide it into chunks (views) that each address a set of related stakeholder concerns. In the fourth document two parts could be distinguished: one part that was relevant to all formulated concerns and one part that was relevant to a specific subset of the concerns. This is interesting, but still does not amount to concern related views that contain all the information that is necessary to address the concerns in question. So for this document the gap may be not as large, but there still is a gap.

The fact that sometimes only a small part of the information is relevant, and moreover it is scattered, makes us suspect that specific stakeholders have difficulty in finding the information of their interest at all. This is a documentation concern that definitely needs to be addressed. To alleviate this problem authors of architecture documents do well to make their stakeholders and their concerns explicit up front, and organize their documents accordingly, for instance by applying IEEE Std 1471 and creating concern related viewpoints and views. Authors can use similar techniques as we have used to investigate four documents: for a new document an intended outline can be matched against intended stakeholders and their perceived concerns.

Authors who are trained in writing documents with a similar set up as we have investigated will not easily apply IEEE Std 1471 to a new document. That is our explanation of the fact that IEEE Std 1471 has up until now not been frequently applied .

8.1.3 Q3 How to define IEEE Std 1471 viewpoints?

To answer this research question we envisaged, as outlined in chapter 4, a method to define IEEE Std 1471 viewpoints and have put that to the test. The method contains four steps:

step 1) Compile stakeholder profile (including concerns),
step 2) Summarize architecture design,
step 3) Analyze the relation between the stakeholder profile and the architecture design,
step 4) Define viewpoints.

For steps 2) and 3) there is a textual variant and a graphical variant. For each step we have developed some tools (templates MS Word or MS Visio).

From working with the participants in the tests of this method we conclude that application of the method takes about half a day to a full day. Compared to the whole effort of producing final documentation, this is not much.

The testers reported a number of benefits, like that it was helpful in imagining the stakeholders and that it was helpful in coming to a shared perception within the design team. Application of the method is beneficial for expressing the positions of the various stakeholders, staying focused on the concerns of the stakeholders, reaching a higher abstraction level in the architectural design and for making clear what design information is needed to address which concerns. Nevertheless, the definition of viewpoints remained somewhat unsatisfactory. The testers reported a lack of support to go from step 3) to step 4).

Our conclusion is that the delineation of views is only partly based on the relation between the architectural information and the concerns of the stakeholders. There must be other factors that come into play. We like to mention here that our comparison of existing architecture frameworks, see chapter 6, revealed another organizing power: the similarity of information. Many frameworks have a major partitioning into types of information, that is, information around similar architectural objects is grouped into separate chapters. This may be to the detriment of the relation to stakeholder concerns, as indicated by our findings in chapter 3, and this in turn may be incompatible with IEEE Std 1471.

More research is needed into the question of what factors determine the partitioning in views.

From the feedback received so far, we conclude that the average practicing IT-architect in ongoing projects prefers to work from IEEE 1471 library viewpoints, instead of creating viewpoints from scratch on his own. We think the reasons for this are the pressure to produce results and a limited ability to reflect on ones work while it is being carried out. Creating new viewpoints is probably more appropriate for people who have time and talent to reflect on architecture descriptions.

8.1.4 Q4 What are the concerns of the stakeholders and which concerns are not addressed by the current architecture documentation practice (for a given architecture department)?

We have undertaken a research activity to answer these questions for an architecture department at a large financial service provider. See chapter 5 for a detailed description. The department maintains a database with information on hundreds of applications and their relations. Architecture studies for selected business domains portray future business functions and the future supporting applications. We devised a list of interview questions and performed a series of 8 in-depth stakeholder interviews to solicit stakeholder concerns and communication preferences. Our main conclusion is that the present setup of an architecture report addresses most concerns that the stakeholders have for this kind of reports, but that additional viewpoints are necessary. The answers of the interviewees confirmed the current work of the architecture department. From the answers we derived three concerns (on Time, Money and People) which were not addressed by the current architecture documentation practice of the department. We also prepared a revision of the used list of interview questions, based on the experiences in this round of interviews, see Appendix B. Working with the list of interview questions proved fruitful: we were able to touch on many areas of concern in a short time and an improvement of the architecture documentation practice was found. We think the revised list of interview questions can be an asset when used as an inquiry tool by architects to evaluate an existing documentation practice of architecture, or to devise a new documentation practice of architecture. Because stakeholder concerns are an essential ingredient of the design process itself, the questions can also be used at the start of an architecture design project. The round of interviews can also be carried out as part of step 1) of our method viewpoint design (creating stakeholder profiles). Carrying out the interviews and processing the results may take up to one or two weeks time. The broad range of questions helps to find relevant concerns that are otherwise easily overlooked. Working with an appropriate, predefined list of questions gives a higher quality to the interviews and makes it easier to combine the results of the interviews.

In this research activity we made an unexpected observation. The stakeholders with no background in architecture were able to articulate (in their own words) the essentials of such an abstract subject as architecture. This is a valuable asset which we believe bears hope for the future communication between stakeholders and architects.

8.1.5 Q5 What can we learn from a comparison of existing architecture frameworks regarding the communication of IT-architecture?

There are many differences between existing architecture frameworks. They all look very different: they have other dimensions, different number of dimensions or different values in equally named dimensions. Partly this can be explained from their original goal, and the context from which they originated. A commonality is that architectural information is often organized in a matrix that is bound by two dimensions: one dimension typically addresses the type of information, and a second one is having a sequential order (order in which architectural information is produced). The second dimension can also be seen as having an increasing level of detail or decreasing scope. For some important concerns, like security, there is sometimes a coordinated view.

We have derived nine base dimensions from the existing architecture frameworks: Type of information, Scope, Detail level, Stakeholder, Transformation, Quality attribute, Meta level, Nature and Representation. These base dimensions can be used apart from architectural frameworks and they allow us to better understand and compare existing frameworks, or to create a new framework. They also ease the understanding and communication of architectural descriptions. See chapter 6 for more details.

There still remains a lot of work to be done in architecture description standardization. In particular, the values within the dimensions described need to be widely agreed upon. This will lead to standardized architectural viewpoints (library viewpoints), and will ultimately contribute to the further maturation of the architect profession.

New frameworks can be created, or existing frameworks can be evaluated, by following these questions: what is the type of architecture information we want to communicate? And, what are the phases in our process of establishing and maintaining architecture? Or, to what levels is our decision taking process geared to? The framework that preeminently parallels the answers to these questions is a good candidate to follow. Choosing an architecture framework is a strategic decision that needs management approval. The lead IT-architect of a company can draw up a proposal for this after consultations with all the stakeholders. Once you have been creating architecture documents that fit in a certain framework, it will be very hard to switch to another framework. We would be hesitant to adjust an existing framework to specific circumstances in one's company, because that makes the documents less recognizable to outsiders and makes available guidelines for the framework less usable. But on the other hand, it is of great value, when a document setup fits like a glove in the development process or decision taking process, and after all most communication is intra-organisational.

8.1.6 Q6 How can we describe Enterprise Architecture in a way that is easy to create (author) and easy to understand (reader)?

Describing Enterprise Architecture (EA) in such a way that it is easy to create and easy to understand is possible by restricting the description to only the most crucial essentials of EA. On the business modeling side this means choosing for the 'enterprise function' as the key descriptive element and treating other information, like flow of information or the process sequence in which business functions are performed, as subordinate. Work processes may change, organization may change, but often the same business functions continue to exist. Business function is a very stable organizational element to base strategic IT-investments on. On the IT-support side a choice is made for the concepts 'computer', 'network' and 'application'. With these key concepts, and only a couple more, a workable EA conceptual model is feasible. Five types of diagrams are necessary to capture the EA essentials:

- the Supply Chain Diagram (shows the cooperation with other enterprises to serve the customers),
- the Enterprise Function Diagram (EFD; top down breakdown of main business functions),
- the Scenario Overlay (work processes on top of EFD models),
- the Application Overlay (applications on top of EFD models) and
- the System Infrastructure Diagram (computers, network connections and applications).

See chapter 7 for the details of this easy EA Modeling method (EAM).

It takes students about 8 hours of study and exercises to learn EAM and the resulting diagrams can be read by a large audience without any formal training. Students, who will restart using EAM later on in their jobs, will easily remember how to do it, because the key concepts are broadly recognized and the diagramming is simple. We perceive the business management as the premier targeted audience for the resulting EAM models.

Some attention points have come out of the usage feedback, which we will deal with in the further development of EAM. For the students the collection and modeling of business information to create EAM diagrams needs further attention, for the readers a reported lack of information needs to be sorted out.

8.1.7 How can the practices of communication of IT-architecture be improved?

Having answered the sub questions we can return to the main question that was the lead motive in our research: How can the practices of communication of IT-architecture be improved?

We would like to summarize our findings in the epitome “meaningful structuring”. We see this sprouting as a recurring theme in our answers to our research questions. With ‘structuring’ we mean actions like grouping information and taking decisions about which information to put at the forefront and which information to give a less prominent position or dropping all together. With ‘meaningful’ we express that the interest of the reader is the driving principle in the structuring. Meaningful also entails that the terminology used, and the labeling of information snippets, must be understandable to the reader and the information must be relevant to his concerns. The order (structure) in which the information is presented must enable the reader to quickly determine which information is relevant to him.

How do we see this “meaningful structuring” recurring in our answers?

In our answer to Q1 the structuring of diagrammatic information is prominent. Which visual attributes enforce the messages we want to get across (what is meaningful to the readers)? Which visual information is distorting the messages and must be dropped? Which visual symbols are meaningful to the reader?

In our answer to Q2 we found that current architecture descriptions have a structure, but that that structure is not very meaningful to the readers, that is, not explicitly linked to their concerns.

As for Q3, based on the relation of the architectural information to the concerns of the stakeholders a structure is sought that is as meaningful to the stakeholders as possible. This structure is expressed in views.

Applying the stakeholder concern inquiry tool, which we constructed in answering Q4, gives full prominence to what is meaningful to the stakeholder. Where do his real concerns lie? Which language does he use?

The comparison of architectural frameworks (Q5) showed meaningful concepts or lists of concepts (dimensions), with which stakeholders are familiar, to structure collections of various information about IT-architecture.

The EAM method (Q6) selects information that is relevant (meaningful) to managers when they take decisions about IT-support for their enterprise functions. This information is structured in five diagram types of which the meaning is easily conceived.

For the details of ‘meaningful structuring for IT-architects’ we refer to the papers in this thesis and the summaries in the previous sections. We now want to go back to the preliminary exploration of Communication in IT-architecture in chapter 1 and see what our findings add to these.

Conclusions of this research

From a general communication perspective (see sections 1.3 and 1.4) we can answer our research question as follows. For the receiving part communication entails these steps: 1) identifying relevant information, 2) reading the information (syntactic level) and mentally constructing the messages carried by the information (semantic level), 3) processing the messages against previous knowledge and making inferences. Identifying relevant information is improved by organizing architecture information in meaningful views (see chapters 3 and 4) and by partitioning large portions of architectural information along well recognized dimensions (see chapter 6). Reading the information in architecture diagrams is improved by carefully using the visual attributes (see chapter 2) and by using relevant and broadly recognized concepts (chapter 7). Processing the messages is improved by offering information that pertains closely to the concerns of the reader (see chapters 4, 5 and 7).

Following the Merriam Webster definition we spoke of communication of IT-architecture as an exchange of information by means of a common set of symbols and signs. What we now can add to that is that this information must be structured in a way that is as much as possible meaningful to the receiver and likewise that symbols and signs (encoding) must be chosen that are meaningful.

In the Shannon-Weaver model, see Figure 1.1, the communication of IT-architecture is a two stage process of encoding by the IT-architect and of decoding by the various stakeholders. The art of encoding mainly consists of making the decoding as smooth as possible. Meaningful structuring, as introduced in this section and worked out in the answers of our research questions, applies very well to the encoding activity of IT-architecture. Since it is impossible to decode a report in one go, the information is broken into pieces (structured) and the pieces are labeled, to enable proper decision taking during the decoding.

The Elaboration Likelihood Model highlights the importance of providing motivational arguments for reading an architecture description. We think this criterion is very well fulfilled when an architect performs an explicit inquiry into the concerns of the stakeholders (Q4) and lets these concerns shine out in the structuring of the information he has to offer.

The Network Theory and Analysis model maintains that the architect preferably should be or become part of the social network of the stakeholders. So we can state that the inquiry into the stakeholder concerns preferably should take place in face to face meetings. Also the presentation of intermediate and final results should take place in face to face meetings and, where possible, should follow company standards of the stakeholders. By using language, colors and diagrammatic symbols (Q1) of the working environment of the stakeholders, the architect in subtle ways shows himself as part of the social network of the stakeholders.

As for the Speech Act model, we already mentioned in section 1.4 that the intentions of the architect should be clear to the stakeholders. Focusing on the concerns of all stakeholders and giving an account of this in the rationale (IEEE Std 1471) makes the work of the architect very transparent and trustworthy. We found the ‘concern’ concept of IEEE Std 1471 a very worthwhile concept to elaborate on, and beneficial to achieve meaningful communication. Making concerns explicit is a rational and practical way of dealing with stakeholders in a IT-architecture design project.

Inspired by Eppler (2004) we can say that the communication of IT-architecture has three major issues: first, reducing or synthesizing insights adequately, second, adapting these trimmed insights to the stakeholder context without distorting them, and third, presenting the compressed and adapted findings in a trust-building style. Again, here our meaningful structuring fits very well. By focusing on stakeholder concerns and by offering guidelines for graphics and (enterprise architecture) views (Q1, Q4 and Q6), this research helps IT-architects in synthesizing their knowledge and adapting it to the stakeholder context.

8.2 IEEE Std 1471 conclusions

In this section we summarize our findings concerning IEEE Std 1471 (IEEE, 2000) as stated in the papers (see chapters 3, 4 and 6). We have offered these findings to the working group that is taking care of the revision of IEEE Std 1471.

These are the attention points concerning IEEE Std 1471 which we derive from our research:

- extend IEEE Std 1471 with guidance to achieve qualities like *accessibility* and *understandability*
- formulate a broad expectation regarding *quantifications*
- point to more factors to *delineate* architectural views
- propose or prescribe *specific viewpoints* (frameworks)
- clarify compliancy or not of existing *frameworks*

We will now show how we came to these attention points and explain them a bit more.

On our first acquaintance with IEEE Std 1471 we tried to imagine the benefits of this standard and how an IEEE Std 1471 compliant document would look like.

With respect to our interest in communication of architecture, the main contribution of IEEE 1471 is the explicit orientation on stakeholders and concerns. Following the path from his recognized concerns via the prescriptions in the viewpoint a stakeholder should be able to find the information of his interest in the views. For the architects IEEE Std 1471 is a stimulus to be very conscious of the concerns of the stakeholders and this helps him to shift away from a possibly too big emphasis on technical aspects.

Having said this, we still feel that the standard lacks vision on effective communication. For instance, being able to find all the information may in practice mean not finding the information. If a stakeholder has to refer to many different views in an architectural document and has to assemble a coherent picture of his concerns by himself, this may in practice mean that he will not go after this information and try to live with his best guesses. The same goes for information that is stated in terms or diagrammic presentations that the stakeholder is not familiar with. We feel the IEEE Std 1471 should be extended with guidance to achieving qualities like *accessibility* and *understandability*.

Another thing to point at is the lack of *quantitive* information in the standard. This applies to sample data from real-life architecture descriptions that follow the standard, as well as to expectations about the future architectural descriptions that will be compiled following this standard. Do the authors expect on average 5 viewpoints in an architecture description, or 25, or 100? How many concerns does a stakeholder have? How many viewpoints are necessary to address a concern? Not formulating at least an expectation leaves too much room open for interpretation of the standard.

From the unexpected remaining gap between step 3 and step 4 of our method viewpoint design (see chapter 4) we conclude that the orientation on stakeholders and their concerns, as prescribed by the IEEE Std 1471 conceptual model, is in itself insufficient to *delineate* architectural views. More factors have to be taken into account, like, for instance, the inherent relations between the key architectural concepts as expressed in the concept map of the method, or the similarity (type) of information.

Although IEEE 1471 is an important contribution to standardising architecture terminology, it still leaves a number of things unspecified. Most importantly, IEEE 1471 does not propose nor prescribe any *specific viewpoint*, which might confuse architects and stakeholders. In a specific context, two architects can easily disagree on who the stakeholders and their concerns are, and what information is needed to address these concerns. Also, if a view contains a lot of architectural information, it needs to be structured, bringing back the needs for which frameworks have been defined. This also holds at the enterprise level, where many IEEE 1471 compliant architectural descriptions may need to be made accessible. So, even if all IT architects would follow IEEE 1471, architectural information could still be very different up to a point where documents are still not accessible, nor comparable. Specific viewpoints, or library viewpoints, should be defined to be used in often recurring circumstances. The "dimension" concept provides a means to further structure large IEEE 1471 views into more manageable chunks.

We see a mismatch between IEEE 1471 and existing architecture practice as represented by the *frameworks* in our overview (see chapter 6). IEEE 1471 requires a view to address a set of related concerns. The "chunks" in which existing frameworks divide the architectural information are addressing many concerns, but it is not obvious these concerns are "related" in the sense of IEEE 1471. Our guess is they aren't, but a difficulty here is that IEEE 1471 does not specify what "related" exactly means.

8.3 Contributions from this research

This section lists the contributions of this research. Apart from gaining insight we have added various tools to the toolkit of the IT-architect:

- 158 Guidelines for improving the readability of IT-architecture diagrams, see chapter 2 and appendix A. More guidelines were found, but these guidelines received the support of the participating IT-architects.
- A technique to quickly relate the content of architecture reports to concerns of stakeholders, see chapter 3. For a given IT-architecture report this technique can be used to evaluate the accessibility of the report. It can also be applied to the outline of reports to be.
- A viewpoint design method to define IEEE Std 1471 viewpoints. The method can be used in an architecture project to prepare a communication outing to the stakeholders. The method can be particularly used to convert existing non-IEEE Std 1471 documentation into IEEE Std 1471 compliant documentation. See section 4.3. See also the future work section.
- An inquiry tool to solicit stakeholder concerns. See chapter 5 and appendix B. The tool is a varied list of architecture related questions that stimulate a stakeholder to express his concerns. The tool can be used to evaluate an existing architecture documentation practice and see what additional viewpoints are necessary.
- A handy overview of the many dimensions used in existing architectural frameworks, see section 6.2.2. This overview can be used (and already is used) as reference material in courses on IT-architecture.

- Nine architecture base dimensions that can be used independently from architectural frameworks to position individual architecture documents, or construct new frameworks. See section 6.3.1.
- A description of a modern lightweight Enterprise Architecture Modeling method. See chapter 7. The method provides an easy way to do a first analysis of an enterprise and its IT-support. The resulting diagrams are readable by a broad public.
- We have gained practical experience with the conceptual model of IEEE Std 1471 and formulated attention points which can be used in the review of this standard. See section 8.2. See also the future work section of this chapter.

8.4 Reflection on research methodology

As said in the introduction our research has been carried out for a large part in cooperation with practicing IT-architects. We have truly appreciated this way of working. Listening to and discussing with practitioners is a very good way to expand one's knowledge and to create new concepts and ideas. Producing methods and tools as tangible results that can be readily applied in practice is very motivating. This does not mean we don't have wishes for improvements concerning any possible future Action Research activities.

The first characteristic of Action Research according to Davison et al. (2004) is the Researcher Client Agreement. We have invited companies to participate in our PhD research plan and three companies responded positively. But along the way we assumed too easily that the companies understood the motives and way of doing of the various research activities. There were moments where the researcher was very busy, but the participating practitioners were not wholeheartedly involved, without us noticing this. In line with this we feel we should have done more Action Research 'diagnosing' activities in cooperation with practitioners to arrive at a really shared common problem statement.

What we found particularly challenging about Action Research was establishing the link to existing theory. Because the trigger for an action research activity is found in a practical situation there is not necessarily a body of knowledge with which the situation can be modeled and with which further research questions can be formulated. In reporting back to the scientific community it can be difficult to provide the 'normal' amount of links to related work. You are in fact out on your own and need a pioneering mentality. It gives all the more a feeling of reward when the research results are recognized in peer reviews as valuable for the scientific community. So this has been challenging, but it is ok.

8.5 Future work

Our research into the communication of IT-architecture has not only given us answers to questions. It has also given rise to many new and interesting questions. We could easily and joyfully spend many more years in research… But that would probably again result in even more unanswered questions… In this section we would like to name the main promising threads for further research we see.

8.5.1 Structuring Issues

The structuring aspect of communication of IT-architecture has come back in our research mainly in the structuring of visual components of diagrams and in the structuring of (families of) architectural documents as a whole. We derive these interesting points for future research:

- Further improving the readability of architectural diagrams
  We have collected 158 guidelines for the readability of IT-architecture diagrams. The

guidelines are grouped into sections: hierarchy, colours, etc. We would like to know more about these guidelines and how they contribute to creating visual structures. Which guidelines take preference? What exactly is their contribution to the readability of diagrams? Which guidelines are easy to apply, which are more difficult? Which guidelines are mutually related? Which guidelines are most sinned against?

- More research into the factors that drive the delineation of views (conflict in structuring forces)
  From IEEE Std 1471 we have taken the notion that a view contains the information that addresses a related set of concerns. But having outlined the concerns of stakeholders it still proved difficult to define the views. We encountered another approach while compiling the overview of architectural frameworks. We saw many frameworks that had views which brought together all the information regarding one key architectural concept. How exactly do these two approaches relate to each other? Are they reinforcing each other or are they conflicting? How do the 'views' in existing frameworks relate to the concerns of the stakeholders? How easy can the stakeholders find the information that is of concern to them in the existing frameworks? What possible other forces induce the delineation of architectural views?
- Collecting library viewpoints (know more about viewpoints in practice)
  IEEE Std 1471 states that views should be made after the prescription in a viewpoint. Wherever IEEE Std 1471 is applied viewpoints exist. We are curious about these viewpoints. Do they exist in practice? What forms do they take? What successful applications of IEEE Std 1471 exist? Some viewpoints are available from books like (Clements at all, 2003), (Lankhorst, 2004) and (Rozanski and Woods, 2005), but are they used in practice? Abridged or unabridged? We think this is a necessary addition to the generic (theoretical) model of IEEE Std 1471.
- Frameworks and dimensions
  The work on architecture description standardization is by no means finished. We have proposed nine base dimensions and would like to get agreement on the values that can be used in these base dimensions. This will be a contribution to the maturation of the architect profession and helps structuring architectural information. The differences between the existing frameworks fascinate us. What is the reason for these differences? Can we explain the differences by looking at the circumstances in which the frameworks are used? Can we develop a theory of frameworks? Give guidance in choosing the right framework for the right situation?

8.5.2 Issues of Meaning

A well known saying is "Beauty is in the eye of the beholder". With a small twist we can say "Meaning is in the eye of the stakeholder". In the end it are the stakeholders who determine the worth (meaning) of a architecture design. As said in 8.1.7 we are very content with the concept of 'concern' as a means of dealing with stakeholders, but the concept lacks a proper foundation. Human communication is part of the 'soft' side of automation. About human communication a lot of research material is available from non-technical environments. In our research we have only made a small beginning in applying these theories to the practical work of IT-architects. Understanding the stakeholder is what is at stake. We derive these interesting points for future research:

- Definition and practical meaning of IEEE Std 1471 concept 'concern'
  The IEEE Std 1471 concept 'concern' has played a vital role in three of our papers. We have investigated the relationship of architecture documents to the concerns of stakeholders. We have proposed a method to outline the concerns of stakeholders in order to define viewpoints. We have constructed an inquiry tool to solicit the concerns

of stakeholders. Since IEEE Std 1471 does not define the term 'concern', nor makes allegations to its practical use, we remained uncertain about staying in or out the intentions of IEEE Std 1471. We would like to produce a definition of 'concern' and give indications of its practical use and offer this to the review body of IEEE Std 1471.

- Learn from requirements engineering
  We want to see what more can be learned from practices in the area of requirements engineering (RE) and apply that to soliciting of stakeholder concerns. Requirements engineering is a more mature strand in IS research, than architecture. What techniques exist in RE that could be applied to architecture, to even better understand the needs of stakeholders of architecture design projects?
- Explore social and psychological aspects of communication of IT-architecture
  In the social sciences various theoretical models are proposed to explain communication. From these models attention points can be derived that were not treated in this research. Matters such as the social context in which the message is transmitted, the assumptions made by source and receiver, their past experiences, distraction, differences in the use of the code, emphasising the wrong part of the message, attitude towards the sender, attitude towards the message. What meaning is given to a message is in the end a subtle and very personal consideration, based on many social and psychological factors. The architect must early on provide motivational arguments for his proposal (why is it important to read the design and think about it?). Should the architect become fully part of the social network of the stakeholders? Is the architect really clear about his intentions?
  Translating future findings in this area to practical recommendations for IT-architects can make the communication of IT-architecture even more effective.
- Explore the processes in strategic decision taking about IT-support for business functions
  We say this with an eye on further developing the Enterprise Architecture Modeling method (EAM), see chapter 7. We think there is room for improvement of EAM. EAM could be extended by introducing a high level form of data modelling. The diagram types of EAM can be documented as IEEE Std 1471 viewpoints. A tool support is needed. But most importantly, the practical use in actual decision taking processes in real life companies can be investigated in more depth. When the architectural concepts of EAM are indeed meaningful to decision takers, then what exactly is their meaning? How can this meaning be enforced? What other concepts might be useful for decision takers?
  Some practical matters also need further attention. The students give low grades to the creation of Enterprise Function Diagrams, what exactly is their problem? The recipients of EAM-documents give low grades to Supply Chain Diagrams, what exactly is their problem?

8.6 References

Clements, P., Bachman, F., Bass, L., Garlan, D., Ivers, J., Little, R., Nord, R., Stafford, J. 2003. Documenting Software Architectures: Views and Beyond¸ Addison-Wesley, Boston.

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

Eppler, M.J., (2004) Knowledge Communication Problems between Experts and Managers. An Analysis of Knowledge Transfer in Decision Processes, ICA Working Paper #1/2004, University of Lugano, Lugano

Greefhorst, D, Koning, H, Van Vliet, H, 2006. The many faces of architectural descriptions, the journal of Information Systems Frontiers, volume 8 number 2, pp. 103-113.

IEEE, 2000. IEEE recommended practice for architecture description, IEEE Standard 1471.

Koning, H., Dormann, C., van Vliet, H., 2002. Practical Guidelines for the Readability of IT-architecture Diagrams, in: Proceedings ACM SIGDOC 2002, pp 90-99.

Koning, H., van Vliet, H., 2006a. Real-life IT architecture design reports and their relation to IEEE Std 1471 stakeholders and concerns, Automated Software Engineering (ASE) Journal, Volume 13, Issue 2, pp. 201-223.

Koning, H, van Vliet, H, 2006b. A Method for Defining IEEE Std 1471 Viewpoints, the Journal of Systems and Software, volume 79 no 1, pp. 120-131.

Koning, H, Bos, R, Brinkkemper, S, 2006. An Inquiry Tool for Stakeholder Concerns of Architectural Viewpoints: a Case Study at a large Financial Service Provider, TEAR2006 workshop of the 10th IEEE International Enterprise Distributed Object Computing Conference (EDOC 2006) ,16-20 October 2006, Hong Kong, China.

Koning, H, Bos, R, Brinkkemper, S, 2008. A Lightweight Method for the Modeling of Enterprise Architectures: Introduction and Usage Feedback, submitted for publication.

Lankhorst, M. et al, 2005. Enterprise Architecture at Work, Springer-Verlag Berlin.

Rozanski, N , Woods, E, 2005. Software Systems Architecture: Working With Stakeholders Using Viewpoints and Perspectives, Addison-Wesley Professional.

9 Appendix A - Guidelines Readability

9.1 Introduction

This document contains guidelines to enhance the readability (legibility) of IT-architecture diagrams. It is part of the research project about "The visualization of IT-architectures".

Many aspects are related to the visual perception and mental understanding of diagrams. Much interesting research is being done in this field. This research is limited to aspects that can directly contribute to the quality of work of IT-architects.

The emphasis of this document is on the outward appearance of it-architecture diagrams in general. Issues concerning specific types of diagrams will be dealt with in the next research phase about Content & Representation. See the research-proposal for an overview of the project.

The draft version of this document has been evaluated by three groups of practicing IT-architects. Their responses were gathered by means of a questionnaire. As a result some 30 guidelines which did not receive enough support were removed.

Many of these guidelines are selected from the book "Illustrating Computer Documentation" by William Horton, 1991, John Wiley & Sons. The selection has been made by keeping the work of IT-architects in mind. Other guidelines were recorded during conversations with IT-architects in the past months or were derived from personal observations as IT-architect or from other literature.

The guidelines are structured around easy recognizable visual attributes: layout, color, etc. See chapter 1. The order of the attributes is random.

Chapter 2 with Design considerations was later added in an attempt to get some grip on the setting in which diagrams are produced.

I thank Walter Bakker for his contribution to the illustrations at the beginning of each section. I thank Joost Boerstoel, Jan Campschroer, René de Gier, Arthur van der Krabben, Johan van Maris and Hans van Vliet for their comments and remarks, which have added to the content of this document.

This document is accompanied by a document "guidelines readability example diagrams from the internet", which contain an annotated set of diagrams.

Any comments, additions, suggestions for the structure, ideas about how to measure the effect are welcome and can be sent to H.Koning, email h.koning@cs.uu.nl.

The homepage of this research is http://www.cs.uu.nl/people/koningh.

9.2 Guidelines about visual attributes of IT-architecture diagrams

9.2.1 Hierarchy/Focus

9.2.1.1 Layers!

9.2.1.1.1 Design the graphic so it can be read in 30 seconds, in 3 minutes, and in 30 minutes (Horton, 1991, p. 80)

9.2.1.1.2 Graphics must immediately and automatically make the most important point, then present secondary points, and with study reveal details. They must organize information into a clear visual hierarchy. See also 9.2.1.2.1. (Horton, 1991, p. 83)

".. three distinct layers …"

9.2.1.1.3 few online documents can in display have more than *three levels of detail and even this requires using color and varying degrees of brightness (HK: use three levels of detail for large diagrams) (Horton, 1991, p. 183)*

9.2.1.1.4 Both the architecture and the diagram (set) have a hierarchic ordering. These two hierarchies, if they are not the same, must be reconciled.

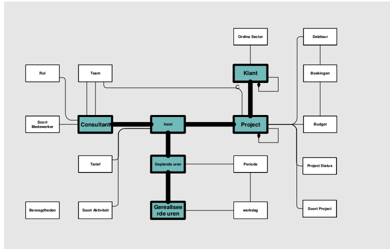


Figure 9-1 original diagram

9.2.1.2 All visual attributes play a role in indicating the hierarchy.

9.2.1.2.1 (A diagram should have) design distinct levels of emphasis, clearly recognizable (size, color, brightness, position, …). See also 9.2.1.1.2. (Horton, 1991, p. 84)

9.2.1.2.2 Lines: heavier for more important relationships. See Figure 9-1. (Horton, 1991, p. 146)

9.2.1.2.3 emphasize primary pattern (thicker lines, short connections) (Horton, 1991, p. 148)

9.2.1.2.4 Don't use more than three distinct levels for a visual attribute (size, color, shade, etc.) (Horton, 1991, p. 179)

9.2.1.2.5 Use primary colors to draw attention to objects that need immediate action. These objects form a distinct level in the hierarchy of the diagram. 'Highlighted representation' means colors that are clearly distinctive from the background.

9.2.1.2.6 The content of a diagram must be clearly distinct from the background and from annotation. Annotation must be less prominent then the content.

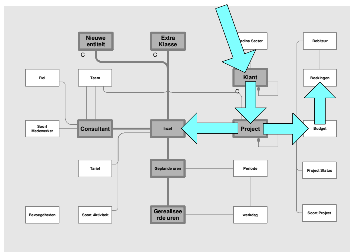


Figure 9-2 diagram dimmed with additional details (access path)

9.2.1.3 Ideas

9.2.1.3.1 Idea: repeat a former diagram in total as a dimmed background in a new diagram to show how the new information relates to the already presented information. This technique can also be used repeatedly to explain additional details or related aspects using an established base of understanding. See Figure 9-1 and Figure 9-2.

9.2.2 Forms of objects, size, width

9.2.2.1 Forms

9.2.2.1.1 A diagram is clear when the outward forms of objects matches the intrinsic properties of the objects (HK: for example, the cylinder shape for a data-storage, which originally resembled a disk-platter.)

9.2.2.1.2 Use more detailed, realistic, three-dimensional symbols for concrete and tangible objects and use simple, geometric shapes for abstract concepts and software constructs. (HK: for example, graphics of computers to denote a hardware platform.) (Horton, 1991, p. 146)

" … intrinsic properties …"

9.2.2.1.3 More than five or six different basic symbols is too many (Horton, 1991, p. 146)

9.2.2.2 Size, width

9.2.2.2.1 Equal size for similar objects (deviations should be meaningful)

9.2.2.2.2 Avoid resizing objects because of short or long texts. For long texts the alternative is to put a short label in the object and put the text in an annotation (insert in the diagram) or in a textbox in the margin of the diagram.

9.2.2.2.3 *Avoid making objects smaller to have more information in one diagram, or making objects bigger to fill up a diagram that otherwise looks to empty.*

9.2.3 Layout

9.2.3.1 Alignment

9.2.3.1.1 *In a series of diagrams similar objects should have a similar position and appearance.*

9.2.3.1.2 *Objects should be positioned on horizontal and vertical lines, e.g. not positioning them so should be meaningful.*

9.2.3.1.3 *Group similar lines or objects (alignment/distance) (Horton, 1991, p. 148)*

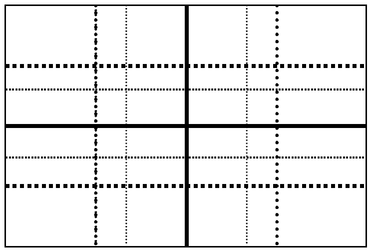


Figure 9-3 Natural positions

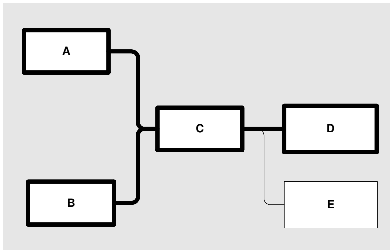
9.2.3.2 Natural positions

9.2.3.2.1 *Meanings which can be attached to vertical positioning:*

9.2.3.2.1.1 *'The higher, the more important' (one of the interpretation-rule which may apply)*

9.2.3.2.1.2 *The higher positioned object has control over the lower positioned object*

9.2.3.2.1.3 *The higher positioned object makes use of the lower positioned object*



no symmetry over horizontal axis

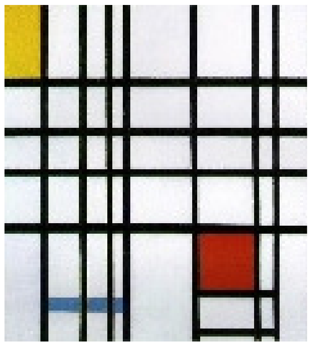
9.2.3.2.2 *Meanings which can be attached to horizontal positioning:*

9.2.3.2.2.1 *The object on the right is 'active' after the object on the left*

9.2.3.2.3 *In positioning objects 'natural positions should be preferred, e.g. central horizontal and vertical axis, secondary axis on ¼ and ¾ or 1/3 and 2/3 etc. See Figure 9-3.*

9.2.3.2.4 *Relations to objects outside the scope of the diagram should be clearly recognizable at the border of the diagram, for instance all input on the left side, all output on the right side.*

9.2.3.2.5 *On an empty diagram the geometrical center is 45% from the top. If you put some shapes on an empty diagram, the diagram looks balanced if the 'center of gravity' of the shapes is 45% from the top and in the middle from left to right. (Horton, 1991, p. 90)*



" … a clear recognizable pattern …"

9.2.3.2.6 *For a stable display, create a bottom-heavy image ('architectural form'). (HK: meaning: more and/or bigger objects in bottom part of diagram) (Horton, 1991, p. 90)*

9.2.3.3 Spacing

9.2.3.3.1 *Keep enough white space on the diagram for annotations, temporary additions, etc. The combined surface of the single objects is not more than 20% of the total surface.*

9.2.3.3.2 *Distances should be equal (deviations should be meaningful, like indicating a grouping)*

9.2.3.3.3 *Spacing between similar objects should be even.*

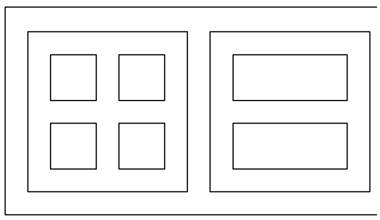


Figure 9-4 chinese box pattern, equal border distance at all levels

9.2.3.3.4 *Enough white space should be provided to distinguish objects, borders, groups, etc. White space is very functional in reading.*

9.2.3.3.5 *In a 'Chinese box' pattern: keep different border distances in the various levels. Compare Figure 9-4 and Figure 9-5.*

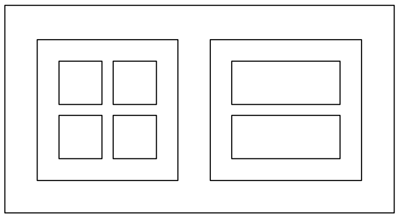


Figure 9-5 chinese box pattern, smaller border distance at inner levels

9.2.3.4 Pattern and flow

9.2.3.4.1 *Choose a clear, recognizable positioning pattern for the objects in a diagram. (Horton, 1991, p. 86)*

9.2.3.4.2 *familiar layout patterns: chain, grid, tree, web (HK: this can be applied to the whole diagram or only for the main (and emphasized) objects) (Horton, 1991, p. 134-135, 144)*

9.2.3.4.3 *On rectangular objects: natural input is left and top, output is right and bottom*

9.2.3.4.4 *The natural starting point is upper left. It should be clear 'what comes next'. (Horton, 1991, p. 81)*

9.2.3.4.5 *Natural: left -> right, top -> bottom. (Horton, 1991, p. 148)*

9.2.3.4.6 *Highly interrelated: objects in circle and straight lines. See Figure 9-6. (Horton, 1991, p. 148)*

9.2.3.4.7 *Arrangement of objects: analogue real world, according "semantic distance", and more general/powerful at top, central concepts in center. (Horton, 1991, p. 147)*

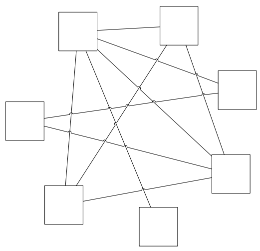


Figure 9-6 highly interrelated

9.2.3.5 Area's

9.2.3.5.1 *Screen layout (GUI): divide the screen in a limited number of clearly recognizable rectangular areas. Functions: overview, detail information, command-buttons, and messages. (Harst, 1999, par 4.7)*

9.2.4 Color

9.2.4.1 Color in general

9.2.4.1.1 *Color is a very powerful visual attribute! Maybe it is the most powerful attribute!*

9.2.4.1.2 *Use color with constraint! (Martin, 1985, p 368)*

9.2.4.1.3 *Light & clear colors get emphasis; keep the rest dull & weak; colors must serve a purpose; subtle, complementary colors are better than strong colors (like pure red). (Harst, 1999, par 5.7)*

9.2.4.1.4 *Colors make a vivid impression, which stays with the viewer while looking at other diagrams.*

9.2.4.1.5 *avoid white objects on a colored area on a white background (this may look like a cheese with holes in it)*

9.2.4.2 application of color

" … color is a powerful attribute …"

9.2.4.2.1 *Highlight importance/draw attention / separate comments from substance of diagram / distinguish types or sorts / simplify by indicating selections / separate overlays / … (Martin, 1985, p 368)*

9.2.4.2.2 *Categorize objects, assign a different color for each; use up to six or seven colors. (Horton, 1991, p. 232)*

9.2.4.2.3 *Applications of color: draw attention,*

Figure 9-7 color sample

group/organize, give status, show relationship. (Harst, 1999, par 5.7)

9.2.4.2.4 *Give text a color that is clearly distinguished from the color of the background. (Harst, 1999, par 5.7)*

9.2.4.3 Choosing colors

Figure 9-8 simple colors

9.2.4.3.1 *Use 'soft' colors. A soft color doesn't yell for attention, and doesn't rival with other visual elements (form, text). A soft color is a mixture of the primary colors (blue, red, green, and yellow) with a light tone. A hard (vivid) color is unmixed like clear blue, red, green or yellow.*

9.2.4.3.2 *Simply using some available colors can lead to cheap effects, and can make a childish impression. See Figure 9-8.*

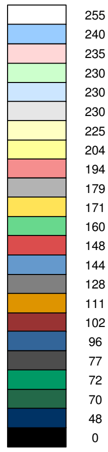


Figure 9-9 color sorted on lightness

9.2.4.3.3 *Use few colors (for instance restricted to the number of main concepts of the user domain, or of the proposed architecture)*

9.2.4.3.4 *For each color-meaning reserve also two added shades: one for subdued/dimmed representation, and one for highlighted presentation.*

9.2.4.3.5 *Use vivid colors only for signaling.*

9.2.4.3.6 *For colors the 'proximity' gestalt-rule applies. So, proximity in color should parallel proximity in meaning. This extends over diagram borders.*

9.2.4.3.7 *Let commonalties in color coincide with commonality in meaning. (Harst, 1999, par 5.7)*

9.2.4.3.8 *Design the use of colors for the whole set of diagrams.*

9.2.4.3.9 *A white background is in fact a vivid color, which tends to provoke other vivid colors (like pure black). So consider using off-white or soft colored background.*

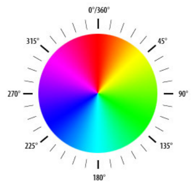


Figure 9-10 Color wheel

9.2.4.3.10 *Colors with opposite hues on the color wheel usually don't match. See Figure 9-10.*

9.2.4.3.11 *A safe rule is: choose light, non primary colors with hues from over the whole color wheel with different levels of brightness. See Figure 9-10.*

9.2.4.3.12 *Stay in tune with the company colors and layout (Harst, 1999, par 5.7)*

9.2.4.3.13 *Colors influence each other; use a neutral background color; don't use red next to green; light colored objects appear more near, dark colored objects appear farther away. (Harst, 1999, par 5.7)*

9.2.4.4 Pitfalls

9.2.4.4.1 *Problems with (too much) color: wrong prior associations, distraction, tiresome, less legible, fuzzy, unreliable. (Horton, 1991, p. 225)*

9.2.4.4.2 *Use not more than four colors to indicate meaning (people cannot remember more). (Harst, 1999, par 5.7)*

9.2.4.4.3 *To overcome problems caused by color-blindness: use color as a redundant signal, don't use red vs. green for any critical distinction, use pure blue only as background, use different lightness values. (Horton, 1991, p. 242)*

9.2.4.4.4 *Use colors of different lightness (because of colorblindness and possible use on black/white monitors or printers). See Figure 9-9, which has the same colors as Figure 9-7, but sorted on lightness. 'Lightness' is the 'L' in HSL color-coding. (Harst, 1999, par 5.7)*

9.2.4.4.5 *Color differences are less good discernable on the computer screen; so keep clear difference in case of online viewing. (Horton, 1991, p. 177)*

9.2.4.4.6 *Don't indicate different meaning only by means of color, but also by means of form/size/pattern/etc. (Harst, 1999, par 5.7)*

9.2.4.5 Per diagram

9.2.4.5.1 Per diagram at most one color for 'highlight'.

9.2.4.5.2 Give foreground and background a clear color contrast. (Horton, 1991, p. 180)

9.2.4.6 Color preferences (Dutch 'voork(l)eur')

9.2.4.6.1 Human perception naturally ranks colors along scale of 'lightness' or along scale of 'warmth' (HK: ➔ proximity rule applies along these scales); cool scale: violet, indigo, blue, green, yellow; warm scale: red, orange, yellow; never mix scales; respect ranking where applicable! (Horton, 1991, p. 223, 232)

9.2.4.6.2 Western viewers tend to prefer colors in this order: blue, red, green, purple, orange, yellow. (HK: be cautious with yellow and orange. Use it only in contrast with the other colors). (Horton, 1991, p. 233)

9.2.4.6.3 Stay close to cultural values, for instance, green carries cultural values of 'safety, ordinary, cold, starting up, in operation', blue has meanings like 'information, cold, out of operation', yellow has 'caution, attention' and red 'danger, emergency, warm, halt, in operation'. (Harst, 1999, par 5.7)

9.2.5 Connectors

9.2.5.1 Connectors in general

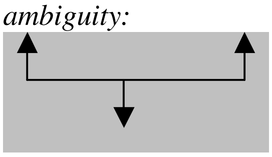
9.2.5.1.1 Lines: solid for primary and concrete relationships, dashed/dotted for secondary of abstract relationships, different patterns for a few different types of relationships. (Horton, 1991, p. 146)

" … connectors …"

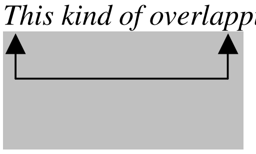
9.2.5.1.2 Is should be clear what the meaning of the arrows is. For instance: in ERD it can mean the 'direction' of a relationship, but it may also indicate a pointer.

9.2.5.1.3 Idea: when there are many connections, show the connections on a separate diagram with the objects simplified and/or reduced in size. After (Horton, 1991, p. 146)

9.2.5.1.4 Let lines overlap, as long as this gives no unclearness of ambiguity. An example of ambiguity:



This kind of overlapping can leave doubt about:



as connection. That is, do the two upper objects also have a relationship to each other?

9.2.5.1.5 When one line crosses another on a diagram without any logical linkage, one of the lines in question will be broken at the crossover. (Martin, 1985, p 368)

9.2.5.2 Endpoints

9.2.5.2.1 Avoid a too great emphasis on line-ends, like arrowheads. If possible, leave the arrowheads out all together.

9.2.5.2.2 Shapes at endpoints of connectors are additional information and should have modest size and 'attention'

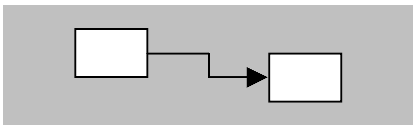
9.2.5.2.3 The information that is represented by the form of line ends can also be represented on detailed diagrams. This gives a less crowded appearance in the more global diagram.

9.2.5.3 Parallel lines

9.2.5.3.1 *Many close parallel lines are difficult to follow individually, A possible solution is maintaining different distances between (sets of) lines.*

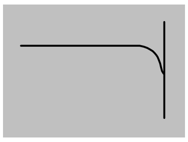
9.2.5.3.2 *If parallel lines bend together, keep equal distances before and after bend*

9.2.5.4 Bends

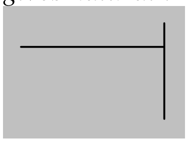
9.2.5.4.1 *Avoid unnecessary bends in connecting lines* 

9.2.5.4.2 *Rounding of bends gives more natural impression of flow*

9.2.5.4.3 *Bend in:*



gives natural impression of direction of flows, over against bump in:

 *(leaves at joining-point 6 directions open)*

9.2.6 Use of text

9.2.6.1 Text in shapes

9.2.6.1.1 *Text in shapes: write in active voice, use action words, use examples, tell what things look/sound/feel/smell, use concrete words that can be memorized verbally and visually. (Horton, 1991, p. 115)*

9.2.6.1.2 *Idea for 'process diagram' with swim lane pattern: put short descriptions in the objects and long descriptions in a separate column (likewise for UML activity diagrams?). (Horton, 1991, p. 122)*

9.2.6.1.3 *If space does not permit a complete label, place a short label in or near the shape and use a footnote to provide more information. (Horton, 1991, p. 146)*

9.2.6.1.4 *Text is a friend of graphics, not an enemy. Well-chosen text can give strong figurative support, and it can make the diagram independently readable. (Horton, 1991, p. 247, 252)*

9.2.6.2 diagram titles and subscripts

9.2.6.2.1 *Provide clear titles. (Horton, 1991, p. 248, 249)*

9.2.6.2.2 *Provide clear subscripts for a diagram (max 3 line). Possible content: position in whole set, importance, reading clue, conclusion, who/what/where/how/why? (Horton, 1991, p. 248, 249)*

9.2.6.2.3 *"There is a difficult choice to make, concerning headings, that is using ordinary-language terms that will be useful to the skimming reader (and in the content list) or using the technical terms which may best describe the topic to be discussed. One solution is to use both a technical title and a non-technical sub-title". (Waller, 1982, p 149)*

9.2.6.3 Annotations

9.2.6.3.1 *Annotations give answers to questions, focuses the attention, and explain. (Horton, 1991, p. 247, 252)*

9.2.6.3.2 *Design labels and annotations so they stand out from the background but remain subordinate to the subject matter. (Horton, 1991, p. 254)*

9.2.6.3.3 *Connect annotation to the object it refers to by means of a short, diagonal, thin line. (Horton, 1991, p. 255, 256)*

9.2.7 Graphics & icons

9.2.7.1 Use of graphics/icons

Figure 9-11 Dynamic images

9.2.7.1.1 Use icons: to speed search, for immediate recognition, for better recall, to save space, for graphic or spatial concepts, for visual appeal, for international audiences. (Horton, 1991, p. 156)

9.2.7.1.2 Use icons and graphics modestly.

9.2.7.1.3 Keep icons and graphics gender neutral and organizational hierarchy neutral.

9.2.7.1.4 Don't try to be funny (unless you are very, very, very sure everybody will understand your good intention).

9.2.7.1.5 Express people values, not 'system' values

9.2.7.1.6 Use graphics to connect associations/emotions (sunny, optimistic, rainy, pessimistic, etc.).

9.2.7.1.7 Use graphics that are meaningful in the daily life of the viewers.

9.2.7.1.8 Dynamic images are more interesting than static images. See Figure 9-11 (Brody, 1982, p 312)

9.2.7.2 Where to get them?

9.2.7.2.1 Start looking for usable graphics during the architecture design project (don't wait until you have to write the final report). Look for graphics which are meaningful to the users/readers and which can help representing the abstract system.

9.2.7.2.2 Search on the Internet in clipart libraries.

9.2.7.2.3 Or make your own (simple) graphics/icons with MS Visio or MS Paint.

9.2.7.2.4 Some guidelines for making your own icons (Harst, 1999, par 5.7):

9.2.7.2.4.1 Stay close to: tasks of the user, the chosen metaphor, context of use, company style, known icons

9.2.7.2.4.2 Maintain consistency (size, color, orientation, light source); start in black & white; add color only for appearance and not for meaning

9.2.8 Diagrams in the architecture report

9.2.8.1 Diagrams

9.2.8.1.1 The distance in the report (number of pages) between a diagram and a reference to it should not be too great (max 2 pages? HK). A possibility is to repeat the diagram (this should be signaled in the subscript). (Horton, 1991, p. 269)

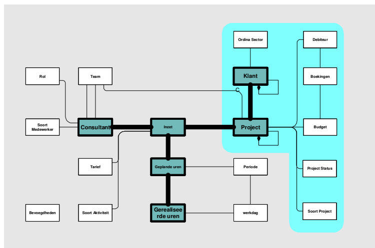


Figure 9-12 thumbnail with focus

9.2.8.1.2 Idea: repeat a thumbnail in the margin of the text. Highlight on the thumbnail the part of the diagram that is being treated in the text. Figure 9-12 could be a repeated thumbnail that accompanies the treatment of the various parts of the depicted model.

9.2.8.1.3 Always number your diagrams (figures). (Horton, 1991, p. 271,272)

9.2.8.1.4 If you place a diagram in the text, provide a clear (visual) distinction: different background color, a line border or horizontal line separators. (Horton, 1991, p. 274)

9.2.8.1.5 For pictures besides text, put the picture to the left and the text to the right. (Hand, 1982, p 105)

9.2.8.2 Text references to diagrams

9.2.8.2.1 Text references to diagrams are always by number (not: above/below, at your left/right). References to diagrams are preferable placed at the end of a paragraph. See to it that all the diagrams are referred to in the text. Don't use diagrams that are not referred to in the text. (Horton, 1991, p. 271,272)

9.2.8.2.2 Provide the reader with … information: What is this a picture of? On what aspects of it should I focus my attention? How will I use it? How is this picture related to what I have been reading? (Brody, 1982, p 306)

9.2.8.2.3 (Brody, 1982, p 307) simply and directly inform the learner of picture content and purpose

9.3 Context and design

In this chapter general remarks are assembled which may give support while designing a set of it-architecture diagrams.

9.3.1 The use of diagrams …

9.3.1.1.1 (Narayanan, 1997, p 15-18) a diagram should support the viewer in reasoning about the subject

9.3.2 Relationship to the organization (client/target)

9.3.2.1 Organization

9.3.2.1.1 Take into account the way of thinking of the organization.

9.3.2.1.2 Show how the system relates to organizational views, workflow views or other views that already exist within the company.

9.3.2.1.3 Stick to the standards.

9.3.2.1.4 Fulfill the expectations of the viewers.

9.3.2.2 Emotion

9.3.2.2.1 Build the viewers confidence: simple; clear visual hierarchy; redundant graphical characteristics; soft edges/round corners; soothing colors. (Horton, 1991, p. 197)

9.3.2.2.2 Win trust: functional graphics; explicit relationships; explicit differences; verifiable. (Horton, 1991, p. 197 – 198)

9.3.2.2.3 The setting in which the text is used is at least equally important to learning as the quality of content and design used to present the information" (emotional safety; clarity of intentions). The personal or organizational circumstances of the reader influence the way he looks at the message of the architect. (Hand, 1982, p 111)

9.3.3 Work process

9.3.3.1.1 The use of diagrams should be designed, just as the architecture itself. It should grow during the design project.

9.3.3.1.2 After using the diagrams for your own design-process, the work begins of preparing the diagrams as a means of communicating the design to the various stakeholders!

9.3.3.1.3 The use of visual attributes should be designed with the whole set of diagrams in mind.

9.3.3.1.4 For each kind of diagram you should have a style that puts the viewer immediately in the right frame of mind to understand the diagram. Style: recognizable combination of visual attributes (color, form, font, etc).

9.3.3.1.5 Composing a diagram: start by identifying the most important information in the diagram. Allow no more than three to seven objects at this top level (see Figure 9-1 for an example). Ideally, identify a single object to dominate the graphic. (Horton, 1991, p. 83)

9.3.3.1.6 After you have prepared the diagram, remove all redundant information. (Horton, 1991, p. 182)

9.3.4 Designing the set of diagrams

9.3.4.1 Structure

9.3.4.1.1 Structuring the information is very important for clear communication. This goes for text, but also for a set of diagrams. If you don't have a clear structure: 'keep moving' until you have it!

9.3.4.1.2 In a clear structure the top level is based on general, familiar concepts (for instance: current situation, proposed new situation, migration). At lower levels, the concepts are more and more specific for the current subject. At all levels the arrangement must be logical (considering the general understanding and the specific issues at stake).

9.3.4.1.3 Give each diagram a small context image (thumbnail; indication of where I am in the total structure of information), which indicates the place of the diagram in the structure of diagrams

9.3.4.1.4 Clues in margin of diagram to: next/previous, up/down?

9.3.4.2 Big things come in small steps …

9.3.4.2.1 "Many small one make one big one …", or, "use small diagrams to introduce the concepts of the big diagrams" (build up in a series of diagrams).

9.3.4.2.2 a complex diagram can be build up in several consecutive diagrams

9.3.4.2.3 movement can be simulated by showing several diagrams with progressive changes in position/state

9.3.4.2.4 Progressively reveal more detailed information (HK: the diagram grows in a series of screens). (Horton, 1991, p. 184)

9.3.4.3 Limits

9.3.4.3.1 Keep in mind the 7 +/- 2 rule for the number of objects which can be seen in one …

9.3.4.3.2 Max 30 objects in one diagram. (Horton, 1991, p. 148)

9.3.4.3.3 Both too much and too little interior detail can reduce the effectiveness of a picture. (Brody, 1982, p 310)

9.3.4.4 Content

9.3.4.4.1 Inserts in the diagram: explanatory text, table of related data, key or legend of symbols, enlargement of a region of dens detail, overview or wide-scale map. (Horton, 1991, p. 246)

9.3.4.4.2 For various purposes (part of) one diagram can be used as a (dimmed) background in another diagram.

9.3.4.4.3 (Dahl, 2000) Two interesting results from experimental research with a group of designers of utensils: the ideas/concepts for new utensils were better when the designer consciously visualized the user together with the utensil and when the designer tried to be imaginative.

9.3.4.4.4 For online reading: design your diagrams with size half A4 (A5), landscape, and put enough information on it to view it without accompanying text. For instance: MS PowerPoint slides.

9.3.5 General questions which people ask themselves while looking at (part of) diagrams

In deciding on the content of a diagram, it may help to realize what a viewer might think while viewing this particular diagram.

9.3.5.1.1 Each graphic should answer these questions: What is it? What is most important? How do I use it? How is it related to other graphics and to the text? (Horton, 1991, p. 87)

9.3.5.1.2 Which objects are on this diagram? What kind of objects? Which additional information is added? Which additional information did I expect?

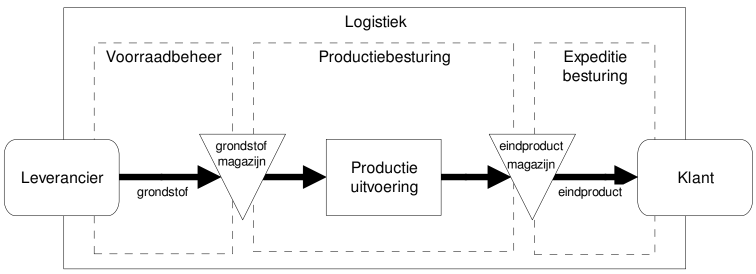
9.3.5.1.3 *Which relationships are there between the objects? What kind of relationships? Did I expect this?*
9.3.5.1.4 *Did I expect this diagram?*
9.3.5.1.5 *Where am I in this diagram? (My interests, my department/relations/work processes)?*
9.3.5.1.6 *What is important? Where does it start? What is the difference compared to the current situation?*
9.3.5.1.7 *How does this relate to the world at large?*
9.3.5.1.8 *What is 'inside', what is 'outside'? (System scope? Company scope? … Scope?)*
9.3.5.1.9 *Where is the value-chain?*
9.3.5.1.10 *Where is the entry-point?*
9.3.5.1.11 *How does this diagram relate to my 'worldview' (conscious or unconscious)(= concepts with which someone holds control over his environment, with which a company is run; like: purchase/sales/production/research/administrative support; "the customer must be served")*

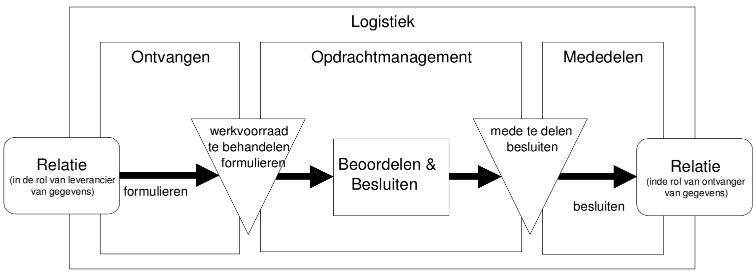
9.4 Examples from workshop-participants

In June 2001 a workshop was held with a group of IT-architects about these guidelines (draft version 0.2). The participants were asked to bring some of their own diagrams. In the workshop they were invited to try some of the guidelines on these diagrams. Here are three before and after images, with some comments referring to the guidelines. The (Dutch) texts have been changed to make the graphics anonymous.

9.4.1 Adding colors

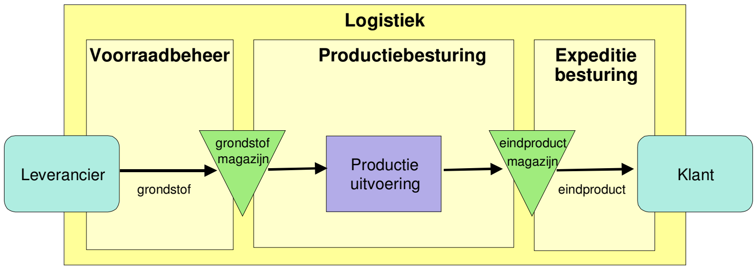
9.4.1.1 Before

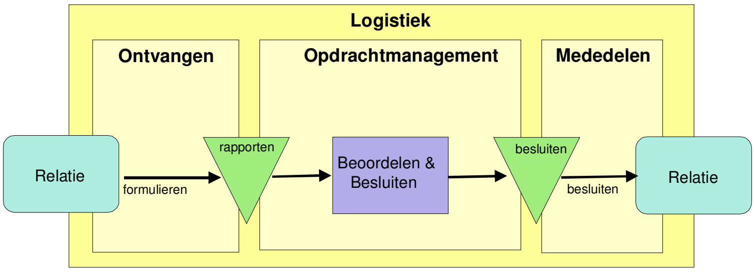




Comments:
- not using colors give nowadays a dull impression
- dominant arrows
- the triangular shapes in the lower part are bigger than in the upper part, but this has no meaning

9.4.1.2 After

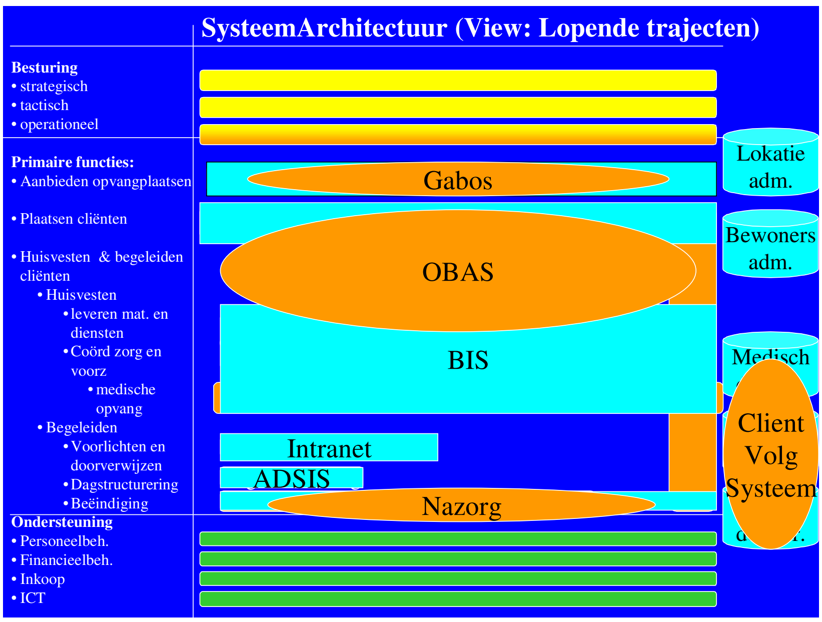




Comments:
- Soft, light colors were chosen
- The two background shades of yellow connect to each other, and are not obtrusive

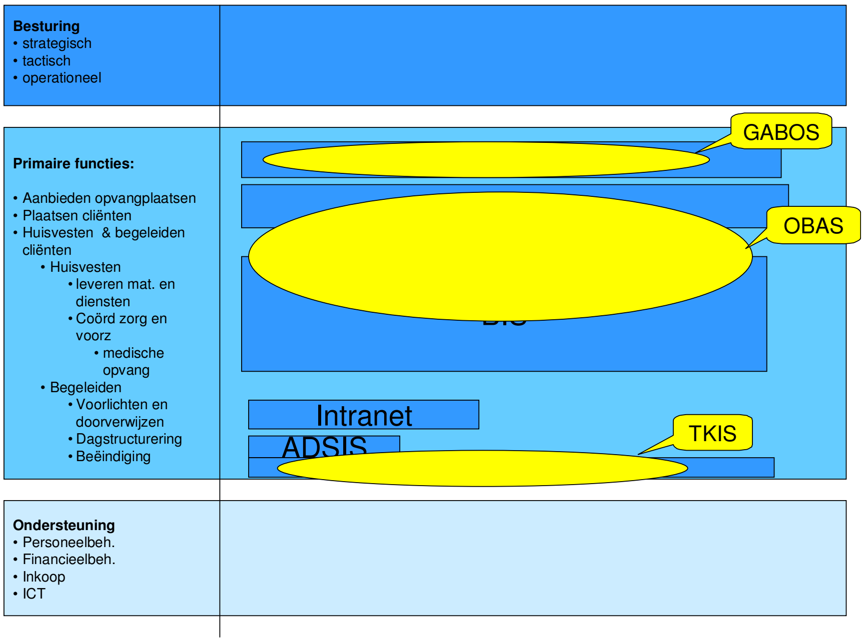
9.4.2 Simplifying content, restraining coloring

9.4.2.1 Before



Comments:
- Crowded
- Orange yelling

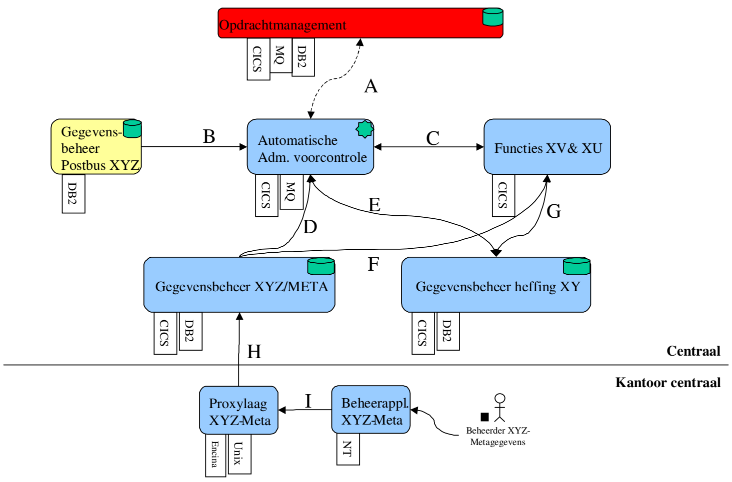
9.4.2.2 After



Comments:
- The basic pattern is maintained which keeps the link to the set of diagrams
- The content is restrained to the purpose of the diagram, further supported by the yellow attention

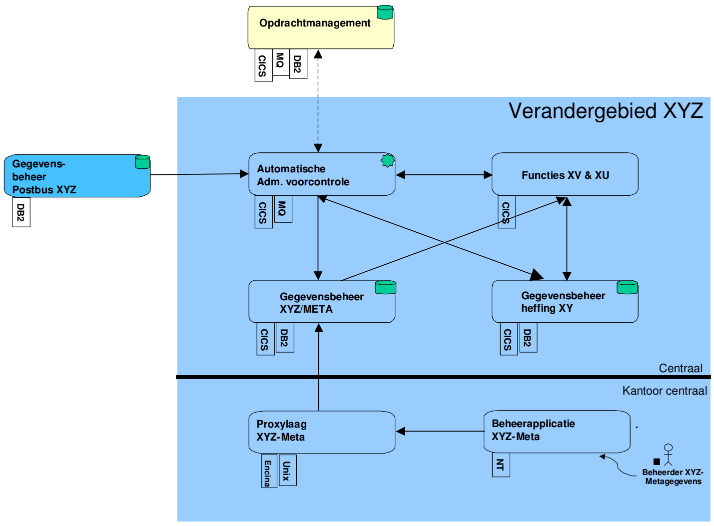
9.4.3 Straightening the pattern and sizes

9.4.3.1 Before



Comments:
- Different sizes and unclear positioning makes the eye wonder around (scanning)
- The red was supposed not to draw attention

9.4.3.2 After



Comment:
- Straightening up gives rest
- The blue shape out of scope is intentionally colored more close to the main figure, than the light yellow

9.5 References

(Brody, 1982) Philip J. Brody, *Affecting Instructional Textbooks Through Pictures*, in 'The Technology of Text', David H. Jonassen ed., Educational Technology Publications, 1982

(Dahl, 2000) Dahl, Chattopadhyay en Gorn, *The importance of visualisation in concept design,* Design Studies, vol 22, no 1, januari 2001, Elsevier

(Hand, 1982) James D. Hand, *Brain functions during learning: implications for text design*, in 'The Technology of Text', David H. Jonassen ed., Educational Technology Publications, 1982

(Harst, 1999) Guido van der Harst, Rob Maijers, *Effectief GUI-ontwerp*, Academic Service, 1999 (in dutch)

(Horton, 1991) W. Horton, *Illustrating Computer Documentation*, Wiley, 1991.

(Martin, 1985) James Martin, Carmen McClure, *Diagramming Techniques for Analysts and Programmers,* Prentice-Hall, 1985

(Narayanan, 1997) N. H. Narayanan, *Diagrammatic communication: A taxonomic overview.* In Kokinov, B. (Ed.), Perspectives on Cognitive Science, Vol. 3, Sofia: New Bulgarian University Press, pp. 91-122.

(Waller, 1982) Robert Waller, *Text as diagram: using typography to improve access and understanding*, in 'The Technology of Text', David H. Jonassen ed., Educational Technology Publications, 1982

(Ware, 2000) Colin Ware, *Information Visualization*, Morgan Kaufmann Publishers, 2000

10 Appendix B – Stakeholder Concern Inquiry Tool

This list of interview questions is equal to the list in the appendix of chapter 5, except that we have omitted the questions that, after 8 interviews, seemed not very fruitful with respect to soliciting concerns or communication preferences.

1. Perspective of interviewee on his own role in the company

1.1. What is the name of your role/function in the company?
1.2. What is the goal of your role/function?
1.3. What are your regular tasks?
1.4. What are the ‘things’ that make up the content of your work? Examples: money, employees, products, …
1.5. What are your worries? What things did you run after, yesterday? Today? Tomorrow? What worries recur regularly?
1.6. What are your plans for the future?

2. Experiences/expectations/relevancy of strategic IT projects

2.1. What does IT mean to you?
2.2. How would you describe what a ‘strategic IT decision’ is?
2.3. Can you give some examples of strategic IT decisions?
2.4. What would be your description of IT-architecture?
2.5. What are the ‘things’ IT-architecture should deal with? Which of these things are captured in IT architecture reports?
2.6. In your experience, under what circumstances is an architecture study needed?
2.7. What do you see as the strong points of the ways in which information technology is used in your company?
2.8. What do you see as the weak points of the ways in which information technology is used in your company?
2.9. What is your definition of a failed IT-project?
2.10. In which ways do you receive information about strategic IT-projects?
2.11. What do you like of each way you receive information about strategic IT-projects, what do you dislike, and why?
2.12. Do you have a wish for the way in which you would like to receive information about strategic IT-projects in future?

3. Wishes future IT-architecture reports

3.1. Suppose one of these days a new project is launched and an architecture study is started to lay the foundation. What questions would you like to see answered by the study?
3.2. At what moments would you like to be involved in the architecture study? For what issues/reasons?
3.3. Wishes with regard to publication media and form?

4. Opinions/wishes about an example architecture report

4.1. Can you please give your own description of what this architecture report is about?
4.2. Suppose you would receive this report today for the first time, for which questions would you seek answers when you would start reading?
4.3. Do you consider this report good readable? Can you explain?

4.4. Do you have any other comment on the setup of this document? (Please don't comment on the content, but only on the way it is presented.)

4.5. Do you agree with the content of the report? Please explain.

11 Nederlandse samenvatting (Summary in Dutch)

Dit hoofdstuk geeft ten bate van familie en bekenden een overzicht van de inhoudelijke resultaten van mijn onderzoek, maar laat ook zien hoe dit allemaal zo gegaan is. Voor verdere details wordt steeds verwezen naar de betreffende hoofdstukken in het boek.

Dit proefschrift bevat de resultaten van enige onderzoeken op het gebied van ‘communicatie van IT-architectuur’. IT-architectuur is voor ons een verzamelnaam om alle vormen van architectuur aan te duiden die je in IT (informatie technologie) tegenkomt, zoals software architectuur, enterprise architectuur, gegevens architectuur, etc. Onder architectuur verstaan wij een verzameling uitspraken die de grote lijnen aangeven en die daarmee een kader vormen voor het verder ontwerpen en bouwen.

Het onderzoek is in part time in 2001 gestart onder eindverantwoording van prof. Hans van Vliet van de Vrije Universiteit, en met steun van drie bedrijven: ING, Ordina en Belastingdienst. Het onderzoek had toen als titel “visualisatie van IT-architectuur”. Het was een combinatie van mijn belangstelling voor beeldtaal in brede zin (ik ben visueel ingesteld) en mijn betrokkenheid bij IT-architectuur als, destijds, IT-consultant. Belastingdienst heeft een jaar meegedaan, Ordina drie jaar en ING vier jaar. Ordina Institute, mijn werkgever in 2001, had een regeling voor het promoveren (de baas betaalt een dag en je betaalt zelf een dag per week). Ook mijn volgende werkgever SERC heeft hieraan meegewerkt.

Mijn interesse voor visualisatie bracht mij als eerste activiteit ertoe om richtlijnen te gaan verzamelen voor het beter leesbaar maken van IT-architectuur diagrammen. Gebaseerd op mijn eigen ervaring als IT-architect heb ik allerlei bronnen doorgekeken op zoek naar richtlijnen: boeken over documenteren van informatiesystemen, boeken over website design, boeken over de menselijke waarneming en over psychologie (Gestalt theorie), artikelen over kleurenleer. Ik heb gesprekken gevoerd met architecten uit de praktijk en hun ideeën genoteerd. Ik heb Internet afgezocht op zoek naar papers en informatieve webpaginas. Uiteindelijk had ik 190 uitspraken verzameld (aanwijzigen: ‘doe dit’, ‘doe dat’) over het maken van diagrammen in IT. De uitspraken gaan o.a. over de bladindeling, over het gebruik van kleuren, over het gebruik van iconen, over het verdelen van de totale informatie in niveaus. Er zijn ook uitspraken over het gebruik van tekst en over hoe je de inhoud van een diagram ontwerpt (wat wil je eigenlijk afbeelden).

Deze uitspraken zijn d.m.v. workshops gepresenteerd aan groepen van praktiserende architecten bij de drie bedrijven die mijn onderzoek steunden. Na drie maanden is een stemronde gehouden, waarbij de architecten per uitspraak aangaven of ze de uitspraak behulpzaam vonden voor hun werk of niet. De lat is gelegd bij 70% steun. Er waren 32 uitspraken die minder dan 70% steun kregen en die zijn verwijderd uit de lijst.

De workshop heb ik later nog enkele malen gehouden, ook op commerciële basis voor heel andere groepen.

Zie hoofdstuk 2 voor de paper over dit onderwerp en bijlage A voor de richtlijnen. Op de website van het onderzoek (http://www.cs.uu.nl/people/koningh) is nog een document te vinden met een verzameling diagrammen die op Internet zijn gevonden en waarbij ik per diagram op basis van de richtlijnen een commentaar geef

In 2002 suggereerde Hans van Vliet dat ik tijdelijk een onderwerp zou doen van algemener belang, om daarna weer terug te keren bij diagrammen. Na veel denkwerk en gebeden ben ik uitgekomen op het IEEE Std 1471. Deze standaard bevat een conceptueel model van een architectuurbeschrijving en leek goede aanknopingspunten te bieden zowel aan de wetenschappelijke kant als aan de praktijk kant. De standaard is populair in kringen van architectuur, maar wordt maar weinig toegepast. Als onderzoeksactiviteit heb ik ervoor gekozen om voor een aantal bestaande documenten, die opgesteld zijn zonder IEEE Std 1471 in acht te nemen, te onderzoeken hoever zij van het model van 1471 af staan. Volgens IEEE Std 1471 bestaat een architectuur beschrijving uit views. Een view is gemaakt aan de hand van een viewpoint. Een viewpoint is dus een voorschrift voor een view. Een viewpoint geeft tevens aan welke belangen/zorgen van belanghebbenden in de view behandeld worden.

Een kernbegrip in IEEE Std 1471 is de concern (zorg of belang) van de stakeholder (belanghebbende). Aan de (dichtbij)auteurs van vier bestaande architectuurrapporten hebben wij gevraagd om aan te geven wie in hun ogen de belangrijkste stakeholders waren van hun document. Vervolgens is gevraagd om van die stakeholders de voornaamste concerns te benoemen en aan te geven waar in het document de informatie staat die die stakeholder nodig heeft betreffende zijn concerns. Zie hoofdstuk 3 voor de tabellen en de conclusies. Belangrijkste conclusies waren: een stakeholder is vaak maar in 25% à 50% van het document geïnteresseerd, en de voor hem relevante informatie zit zeer versnipperd in het document (zozeer dat je jezelf kan afvragen of de informatie wel altijd gevonden wordt). Verder bleek dat een nieuwe ordening van de documenten naar de concerns van de stakeholders niet voorhanden lag en dat de afstand tot IEEE Std 1471 dus groot was.

Dit onderzoek is de inspiratie geworden voor de methode viewpoints design in 2003.

Het feit dat de bestaande architectuurdocumenten niet eenvoudig waren om te sleutelen tot IEEE Std 1471 compliant documenten, bracht mij tot de conclusie dat compliancy met (het voldoen aan) IEEE Std 1471 vanaf het begin van het schrijven van een architectuur document in acht genomen moet worden. Ik heb een methode bedacht voor het definiëren van IEEE Std 1471 viewpoints. Een viewpoint is een voorschrift voor een view. Om betere views te krijgen moet je dus betere viewpoints hebben, en de methode is daarop gericht. De methode bevat vier stappen:

stap 1) opstellen stakeholderprofielen (inclusief concerns),
stap 2) samenvatten architectuurontwerp,
stap 3) analyseren van de relatie tussen 1) en 2), en
stap 4) definiëren viewpoints.

Bij 2) en 3) is een tekstuele variant en een grafische variant. Bij elke stap zijn bescheiden hulpmiddelen (templates MS Word of MS Visio). Er is een traject uitgezet om deze methode te bediscussiëren en uit te testen. Bij twee gelegenheden is de methode door een student of door studentengroepen toegepast. Met architecten uit de praktijk is in workshopverband gediscussieerd en is de methode voor een bestaande situatie (waar al documentatie was) opnieuw toegepast tot het punt van definiëren viewpoints. Van alle sessies heb ik aantekeningen gemaakt en die heb ik zorgvuldig verwerkt en daar aanbevelingen uit gedestilleerd voor verbetering van de methode.

Zie hoofdstuk 4 voor een beschrijving van de methode en de evaluatiepunten.

Ondanks dat er door ‘echte’ architecten nog geen views geschreven zijn volgens d.m.v. deze methode gedefinieerde viewpoints ben ik bemoedigd over de waarde van de methode. Vooral door de spontane reacties van architecten, dat de methode hun aan het denken zet en nieuw inzicht geeft. Rond de zomer van 2003 heb ik een architect van Ordina begeleid bij het toepassen van de methode in zijn situatie. Ook van hem ontving ik enthousiaste reacties.

In 2003 heb ik mij intensief bezig gehouden met de vraag wat voor soort onderzoek ik doe. Ik doe dus ‘kwalitatief onderzoek’, omdat ik op een vrij open manier de belevingen van mensen onderzoek. Binnen kwalitatief onderzoek doe ik de variant ‘action research’, omdat ik ook wat wil veranderen aan de bestaande situatie.

Tevens heb ik de titel van het onderzoek in deze tijd veranderd in ‘communicatie van IT-architectuur’ om zodoende de onderzoekingen rond IEEE Std 1471 er ook onder te vangen. Hiermee was ook de centrale onderzoeksvraag definitief gevestigd: hoe kan de praktijk van het communiceren van IT-architectuur verbeterd worden?

Een van de feedback commentaren uit de evaluatie van de methode viewpoints design was ‘geef ons library viewpoints’ (kant en klare viewpoints). Dit heeft geleid tot het idee om een diepte interview van stakeholders te ontwerpen, wat zou moeten leiden tot het vaststellen van de concerns van de stakeholders en tot het, op basis daarvan, definiëren van library viewpoints. Ik heb samen met mijn contactpersoon bij ING een voorspelling opgesteld van stakeholderprofielen, samenvatting van architectuur, relatie daartussen en viewpoints. Deze ‘voorspelling’ beschrijft eigenlijk de huidige praktijk. Ik heb ook met ervaren architecten uit de praktijk een reviewsessie gehouden van de interviewplannen. Dat was erg nuttig, er zijn goede punten uitgekomen.

Het interview is acht keer afgenomen bij stakeholders van één architectuur afdeling bij ING. De antwoorden van de geïnterviewde stakeholders zijn zeer divers en ik heb een tijd nodig gehad om door de bomen het bos te zien. Bij het schrijven van de paper over de interviews (zie hoofdstuk 5) werd het echter steeds helderder. Ik heb geconcludeerd dat op basis van de interviews de bestaande praktijk op drie punten aangevuld moet worden. Er moeten viewpoints komen voor ‘tijd’, ‘geld’ en ‘mensen’. Zie hoofdstuk 5 voor stukjes beschrijving daarbij.

Tot aan 2004 had ik alles afgestemd met architecten, dit was de eerste keer dat ik voorbij de architect in contact kwam met ‘klanten’ van de architect. Dat vond ik wel spannend. In het algemeen heb ik geen probleem om mij te verstaan met architecten, zeker niet met goede architecten. Hun abstractieniveau en manier van denken spreekt mij aan. De contacten met klanten van de architect zijn mij meegevallen, maar zonder alle voorbereidingen was het niet zo goed gegaan.

In 2004 is ook een artikel over architectuurraamwerken tot stand gekomen in een samenwerking met Danny Greefhorst van IBM en prof. Hans van Vliet. Zie hoofdstuk 6 van het proefschrift. Danny en ik zijn de voornaamste auteurs en hebben gelijk bijgedragen. Danny en ik hadden elkaar ontmoet bij SERC (mijn werkgever in 2002). Danny was als oud-werknemer even terug en we kwamen er al snel achter dat we allebei geïnteresseerd waren in architectuurdimensies. We zijn samen begonnen aan een Nederlandstalig artikel voor het Landelijk Architectuur Congres in 2002. Op voorstel van Hans van Vliet zijn we met z’n drieën begonnen aan een Engelstalige versie voor internationale publicatie.

Het artikel bevat een overzicht van bestaande architectuurraamwerken (indelingen van architectuur documentatie) en de dimensies daarin. Dat overzicht is op zich al heel nuttig en leerzaam. Wat we er zelf uitgehaald hebben zijn negen algemeen herkenbare ‘basis dimensies’, die, los van raamwerken, gebruikt kunnen worden om over architectuur te communiceren.

Mijn inzicht in architectuurraamwerken is zeer verdiept door deze exercities. Mijn uiteindelijke conclusie is dat architectuurviewpoints bepaald worden door een logische indeling van het architectuurontwerp a.d.h.v. ‘typen informatie’. Voorbeelden van typen informatie zijn: bedrijfsprocessen, informatiesystemen, infrastructurele voorzieningen, etc.

In de praktijk krijgen deze typen informatie eigen views. Voor enkele belangrijke concerns, zoals beveiliging, wordt er soms een overkoepelende view samengesteld. Als er veel informatie moet worden vastgelegd, bijvoorbeeld voor een heel bedrijf, dan wordt er een tweede as geconstrueerd met een indeling in meerdere abstractieniveaus.

In 2005 ben ik gaan samenwerken met prof. Sjaak Brinkkemper en dr. Rik Bos, beiden van de Universiteit Utrecht. Sjaak en Rik waren enthousiast over hun Enterprise Architecture Modeling method (EAM). Enterprise architectuur beschrijft de hoofdlijnen van een bedrijf en koppelt deze aan de beschikbare IT-ondersteuning. Ik heb mij door hun enthousiasme laten inspireren en heb meegewerkt aan een revisie van de onderwijsmaterialen over EAM. Ter gelegenheid van een paper met een beknopte beschrijving en motivatie van EAM, heb ik twee enquêtes gehouden, een onder de makers van EAM-rapporten (studenten die de enterprise architectuur van een echt bedrijf hebben beschreven) en een onder lezers van EAM-rapporten (de contactpersonen bij deze echte bedrijven). Uit de enquêtes komen enkele punten naar voren waar EAM nog aandacht behoeft (zie hoofdstuk 7 voor alle details).

Ik vond het heel leuk om deze onderzoekingen te doen. Er zijn nog veel onderwerpen om te bestuderen in het kader van de 'communicatie van IT-architectuur'. Zo kan EAM nog verbeterd worden en beter onderbouwd worden en voorzien worden van een tool (software die de toepassing van EAM ondersteunt). Ik zou graag een vergelijking willen doen van tools die IT-architectuur ondersteunen. Het 'concern' begrip van IEEE Std 1471 moet nodig verder uitgediept worden, als mede de relatie van IEEE Std 1471 viewpoints tot de bestaande architectuur raamwerken. Welke factoren bepalen nou uiteindelijk de grenzen van een view? In de kennisgebieden van requirements engineering en van sociale communicatietheorieën ligt nog veel kennis die van nut kan zijn voor de communicatie van IT-architectuur. Zie paragraaf 8.5 voor een uitgebreide beschrijving van mogelijke toekomstige activiteiten.

12 SIKS Dissertatiereeks

```
====
1998
====

1998-1  Johan van den Akker (CWI)
        DEGAS - An Active, Temporal Database of Autonomous Objects

1998-2  Floris Wiesman (UM)
        Information Retrieval by Graphically Browsing Meta-Information

1998-3  Ans Steuten (TUD)
        A Contribution to the Linguistic Analysis of Business
Conversations
        within the Language/Action Perspective

1998-4  Dennis Breuker (UM)
        Memory versus Search in Games

1998-5  E.W.Oskamp (RUL)
        Computerondersteuning bij Straftoemeting

====
1999
====

1999-1  Mark Sloof (VU)
        Physiology of Quality Change Modelling; Automated modelling of
        Quality Change of Agricultural Products

1999-2  Rob Potharst (EUR)
        Classification using decision trees and neural nets

1999-3  Don Beal (UM)
        The Nature of Minimax Search

1999-4  Jacques Penders (UM)
        The practical Art of Moving Physical Objects

1999-5  Aldo de Moor (KUB)
        Empowering Communities: A Method for the Legitimate User-Driven
        Specification of Network Information Systems

1999-6  Niek J.E. Wijngaards (VU)
        Re-design of compositional systems

1999-7  David Spelt (UT)
        Verification support for object database design

1999-8  Jacques H.J. Lenting (UM)
        Informed Gambling: Conception and Analysis of a Multi-Agent
Mechanism
        for Discrete Reallocation.

====
2000
```

====

2000-1 Frank Niessink (VU)
Perspectives on Improving Software Maintenance

2000-2 Koen Holtman (TUE)
Prototyping of CMS Storage Management

2000-3 Carolien M.T. Metselaar (UVA)
Sociaal-organisatorische gevolgen van kennistechnologie;
een procesbenadering en actorperspectief.

2000-4 Geert de Haan (VU)
ETAG, A Formal Model of Competence Knowledge for User Interface Design

2000-5 Ruud van der Pol (UM)
Knowledge-based Query Formulation in Information Retrieval.

2000-6 Rogier van Eijk (UU)
Programming Languages for Agent Communication

2000-7 Niels Peek (UU)
Decision-theoretic Planning of Clinical Patient Management

2000-8 Veerle Coup, (EUR)
Sensitivity Analyis of Decision-Theoretic Networks

2000-9 Florian Waas (CWI)
Principles of Probabilistic Query Optimization

2000-10 Niels Nes (CWI)
Image Database Management System Design Considerations, Algorithms and Architecture

2000-11 Jonas Karlsson (CWI)
Scalable Distributed Data Structures for Database Management

====
2001
====

2001-1 Silja Renooij (UU)
Qualitative Approaches to Quantifying Probabilistic Networks

2001-2 Koen Hindriks (UU)
Agent Programming Languages: Programming with Mental Models

2001-3 Maarten van Someren (UvA)
Learning as problem solving

2001-4 Evgueni Smirnov (UM)
Conjunctive and Disjunctive Version Spaces with Instance-Based Boundary Sets

2001-5 Jacco van Ossenbruggen (VU)
Processing Structured Hypermedia: A Matter of Style

2001-6 Martijn van Welie (VU)
Task-based User Interface Design

2001-7 Bastiaan Schonhage (VU)
Diva: Architectural Perspectives on Information Visualization

2001-8 Pascal van Eck (VU)
A Compositional Semantic Structure for Multi-Agent Systems Dynamics.

2001-9 Pieter Jan 't Hoen (RUL)
Towards Distributed Development of Large Object-Oriented Models, Views of Packages as Classes

2001-10 Maarten Sierhuis (UvA)
Modeling and Simulating Work Practice
BRAHMS: a multiagent modeling and simulation language for work practice analysis and design

2001-11 Tom M. van Engers (VUA)
Knowledge Management:
The Role of Mental Models in Business Systems Design

====
2002
====

2002-01 Nico Lassing (VU)
Architecture-Level Modifiability Analysis

2002-02 Roelof van Zwol (UT)
Modelling and searching web-based document collections

2002-03 Henk Ernst Blok (UT)
Database Optimization Aspects for Information Retrieval

2002-04 Juan Roberto Castelo Valdueza (UU)
The Discrete Acyclic Digraph Markov Model in Data Mining

2002-05 Radu Serban (VU)
The Private Cyberspace Modeling Electronic
Environments inhabited by Privacy-concerned Agents

2002-06 Laurens Mommers (UL)
Applied legal epistemology; Building a knowledge-based ontology of the legal domain

2002-07 Peter Boncz (CWI)
Monet: A Next-Generation DBMS Kernel For Query-Intensive Applications

2002-08 Jaap Gordijn (VU)
Value Based Requirements Engineering: Exploring Innovative E-Commerce Ideas

2002-09 Willem-Jan van den Heuvel(KUB)

Integrating Modern Business Applications with Objectified Legacy Systems

2002-10 Brian Sheppard (UM)
Towards Perfect Play of Scrabble

2002-11 Wouter C.A. Wijngaards (VU)
Agent Based Modelling of Dynamics: Biological and Organisational Applications

2002-12 Albrecht Schmidt (Uva)
Processing XML in Database Systems

2002-13 Hongjing Wu (TUE)
A Reference Architecture for Adaptive Hypermedia Applications

2002-14 Wieke de Vries (UU)
Agent Interaction: Abstract Approaches to Modelling, Programming and Verifying Multi-Agent Systems

2002-15 Rik Eshuis (UT)
Semantics and Verification of UML Activity Diagrams for Workflow Modelling

2002-16 Pieter van Langen (VU)
The Anatomy of Design: Foundations, Models and Applications

2002-17 Stefan Manegold (UVA)
Understanding, Modeling, and Improving Main-Memory Database Performance

====
2003
====

2003-01
Heiner Stuckenschmidt (VU)
Ontology-Based Information Sharing in Weakly Structured Environments

2003-02
Jan Broersen (VU)
Modal Action Logics for Reasoning About Reactive Systems

2003-03
Martijn Schuemie (TUD)
Human-Computer Interaction and Presence in Virtual Reality Exposure Therapy

2003-04
Milan Petkovic (UT)
Content-Based Video Retrieval Supported by Database Technology

2003-05
Jos Lehmann (UVA)
Causation in Artificial Intelligence and Law - A modelling approach

2003-06

Boris van Schooten (UT)
Development and specification of virtual environments

2003-07
Machiel Jansen (UvA)
Formal Explorations of Knowledge Intensive Tasks

2003-08
Yongping Ran (UM)
Repair Based Scheduling

2003-09
Rens Kortmann (UM)
The resolution of visually guided behaviour

2003-10
Andreas Lincke (UvT)
Electronic Business Negotiation: Some experimental studies on the interaction between medium, innovation context and culture

2003-11
Simon Keizer (UT)
Reasoning under Uncertainty in Natural Language Dialogue using Bayesian Networks

2003-12
Roeland Ordelman (UT)
Dutch speech recognition in multimedia information retrieval

2003-13
Jeroen Donkers (UM)
Nosce Hostem - Searching with Opponent Models

2003-14
Stijn Hoppenbrouwers (KUN)
Freezing Language: Conceptualisation Processes across ICT-Supported Organisations

2003-15
Mathijs de Weerdt (TUD)
Plan Merging in Multi-Agent Systems

2003-16
Menzo Windhouwer (CWI)
Feature Grammar Systems - Incremental Maintenance of Indexes to Digital Media Warehouses

2003-17
David Jansen (UT)
Extensions of Statecharts with Probability, Time, and Stochastic Timing

2003-18
Levente Kocsis (UM)
Learning Search Decisions

====

2004
====

2004-01
Virginia Dignum (UU)
A Model for Organizational Interaction: Based on Agents, Founded in Logic

2004-02
Lai Xu (UvT)
Monitoring Multi-party Contracts for E-business

2004-03
Perry Groot (VU)
A Theoretical and Empirical Analysis of Approximation in Symbolic Problem Solving

2004-04
Chris van Aart (UVA)
Organizational Principles for Multi-Agent Architectures

2004-05
Viara Popova (EUR)
Knowledge discovery and monotonicity

2004-06
Bart-Jan Hommes (TUD)
The Evaluation of Business Process Modeling Techniques

2004-07
Elise Boltjes (UM)
Voorbeeldig onderwijs; voorbeeldgestuurd onderwijs, een opstap naar abstract denken, vooral voor meisjes

2004-08
Joop Verbeek(UM)
Politie en de Nieuwe Internationale Informatiemarkt, Grensregionale politi%le gegevensuitwisseling en digitale expertise

2004-09
Martin Caminada (VU)
For the Sake of the Argument; explorations into argument-based reasoning

2004-10
Suzanne Kabel (UVA)
Knowledge-rich indexing of learning-objects

2004-11
Michel Klein (VU)
Change Management for Distributed Ontologies

2004-12
The Duy Bui (UT)
Creating emotions and facial expressions for embodied agents

2004-13
Wojciech Jamroga (UT)
Using Multiple Models of Reality: On Agents who Know how to Play

2004-14
Paul Harrenstein (UU)
Logic in Conflict. Logical Explorations in Strategic Equilibrium

2004-15
Arno Knobbe (UU)
Multi-Relational Data Mining

2004-16
Federico Divina (VU)
Hybrid Genetic Relational Search for Inductive Learning

2004-17
Mark Winands (UM)
Informed Search in Complex Games

2004-18
Vania Bessa Machado (UvA)
Supporting the Construction of Qualitative Knowledge Models

2004-19
Thijs Westerveld (UT)
Using generative probabilistic models for multimedia retrieval

2004-20
Madelon Evers (Nyenrode)
Learning from Design: facilitating multidisciplinary design teams

====
2005
====

2005-01
Floor Verdenius (UVA)
Methodological Aspects of Designing Induction-Based Applications

2005-02
Erik van der Werf (UM))
AI techniques for the game of Go

2005-03
Franc Grootjen (RUN)
A Pragmatic Approach to the Conceptualisation of Language

2005-04
Nirvana Meratnia (UT)
Towards Database Support for Moving Object data

2005-05
Gabriel Infante-Lopez (UVA)
Two-Level Probabilistic Grammars for Natural Language Parsing

2005-06
Pieter Spronck (UM)
Adaptive Game AI

2005-07
Flavius Frasincar (TUE)
Hypermedia Presentation Generation for Semantic Web Information Systems

2005-08
Richard Vdovjak (TUE)
A Model-driven Approach for Building Distributed Ontology-based Web Applications

2005-09
Jeen Broekstra (VU)
Storage, Querying and Inferencing for Semantic Web Languages

2005-10
Anders Bouwer (UVA)
Explaining Behaviour: Using Qualitative Simulation in Interactive Learning Environments

2005-11
Elth Ogston (VU)
Agent Based Matchmaking and Clustering - A Decentralized Approach to Search

2005-12
Csaba Boer (EUR)
Distributed Simulation in Industry

2005-13
Fred Hamburg (UL)
Een Computermodel voor het Ondersteunen van Euthanasiebeslissingen

2005-14
Borys Omelayenko (VU)
Web-Service configuration on the Semantic Web; Exploring how semantics meets pragmatics

2005-15
Tibor Bosse (VU)
Analysis of the Dynamics of Cognitive Processes

2005-16
Joris Graaumans (UU)
Usability of XML Query Languages

2005-17
Boris Shishkov (TUD)
Software Specification Based on Re-usable Business Components

2005-18
Danielle Sent (UU)
Test-selection strategies for probabilistic networks

2005-19
Michel van Dartel (UM)
Situated Representation

2005-20
Cristina Coteanu (UL)
Cyber Consumer Law, State of the Art and Perspectives

2005-21
Wijnand Derks (UT)
Improving Concurrency and Recovery in Database Systems by Exploiting Application Semantics

====
2006
====

2006-01
Samuil Angelov (TUE)
Foundations of B2B Electronic Contracting

2006-02
Cristina Chisalita (VU)
Contextual issues in the design and use of information technology in organizations

2006-03
Noor Christoph (UVA)
The role of metacognitive skills in learning to solve problems

2006-04
Marta Sabou (VU)
Building Web Service Ontologies

2006-05
Cees Pierik (UU)
Validation Techniques for Object-Oriented Proof Outlines

2006-06
Ziv Baida (VU)
Software-aided Service Bundling - Intelligent Methods & Tools for Graphical Service Modeling

2006-07
Marko Smiljanic (UT)
XML schema matching -- balancing efficiency and effectiveness by means of clustering

2006-08
Eelco Herder (UT)
Forward, Back and Home Again - Analyzing User Behavior on the Web

2006-09
Mohamed Wahdan (UM)
Automatic Formulation of the Auditor's Opinion

2006-10
Ronny Siebes (VU)
Semantic Routing in Peer-to-Peer Systems

2006-11
Joeri van Ruth (UT)
Flattening Queries over Nested Data Types

2006-12

Bert Bongers (VU)
Interactivation - Towards an e-cology of people, our technological environment, and the arts

2006-13
Henk-Jan Lebbink (UU)
Dialogue and Decision Games for Information Exchanging Agents

2006-14
Johan Hoorn (VU)
Software Requirements: Update, Upgrade, Redesign - towards a Theory of Requirements Change

2006-15
Rainer Malik (UU)
CONAN: Text Mining in the Biomedical Domain

2006-16
Carsten Riggelsen (UU)
Approximation Methods for Efficient Learning of Bayesian Networks

2006-17
Stacey Nagata (UU)
User Assistance for Multitasking with Interruptions on a Mobile Device

2006-18
Valentin Zhizhkun (UVA)
Graph transformation for Natural Language Processing

2006-19
Birna van Riemsdijk (UU)
Cognitive Agent Programming: A Semantic Approach

2006-20
Marina Velikova (UvT)
Monotone models for prediction in data mining

2006-21
Bas van Gils (RUN)
Aptness on the Web

2006-22
Paul de Vrieze (RUN)
Fundaments of Adaptive Personalisation

2006-23
Ion Juvina (UU)
Development of Cognitive Model for Navigating on the Web

2006-24
Laura Hollink (VU)
Semantic Annotation for Retrieval of Visual Resources

2006-25
Madalina Drugan (UU)
Conditional log-likelihood MDL and Evolutionary MCMC

2006-26
Vojkan Mihajlovic (UT)

Score Region Algebra: A Flexible Framework for Structured Information Retrieval

2006-27
Stefano Bocconi (CWI)
Vox Populi: generating video documentaries from semantically annotated media repositories

2006-28
Borkur Sigurbjornsson (UVA)
Focused Information Access using XML Element Retrieval

====
2007
====

2007-01
Kees Leune (UvT)
Access Control and Service-Oriented Architectures

2007-02
Wouter Teepe (RUG)
Reconciling Information Exchange and Confidentiality: A Formal Approach

2007-03
Peter Mika (VU)
Social Networks and the Semantic Web

2007-04
Jurriaan van Diggelen (UU)
Achieving Semantic Interoperability in Multi-agent Systems: a dialogue-based approach

2007-05
Bart Schermer (UL)
Software Agents, Surveillance, and the Right to Privacy: a Legislative Framework for Agent-enabled Surveillance

2007-06
Gilad Mishne (UVA)
Applied Text Analytics for Blogs

2007-07
Natasa Jovanovic' (UT)
To Whom It May Concern - Addressee Identification in Face-to-Face Meetings

2007-08
Mark Hoogendoorn (VU)
Modeling of Change in Multi-Agent Organizations

2007-09
David Mobach (VU)
Agent-Based Mediated Service Negotiation

2007-10
Huib Aldewereld (UU)

Autonomy vs. Conformity: an Institutional Perspective on Norms and Protocols

2007-11
Natalia Stash (TUE)
Incorporating Cognitive/Learning Styles in a General-Purpose Adaptive Hypermedia System

2007-12
Marcel van Gerven (RUN)
Bayesian Networks for Clinical Decision Support: A Rational Approach to Dynamic Decision-Making under Uncertainty

2007-13
Rutger Rienks (UT)
Meetings in Smart Environments; Implications of Progressing Technology

2007-14
Niek Bergboer (UM)
Context-Based Image Analysis

2007-15
Joyca Lacroix (UM)
NIM: a Situated Computational Memory Model

2007-16
Davide Grossi (UU)
Designing Invisible Handcuffs. Formal investigations in Institutions and Organizations for Multi-agent Systems

2007-17
Theodore Charitos (UU)
Reasoning with Dynamic Networks in Practice

2007-18
Bart Orriens (UvT)
On the development and management of adaptive business collaborations

2007-19
David Levy (UM)
Intimate relationships with artificial partners

2007-20
Slinger Jansen (UU)
Customer Configuration Updating in a Software Supply Network

2007-21
Karianne Vermaas (UU)
Fast diffusion and broadening use: A research on residential adoption and usage of broadband internet in the Netherlands between 2001 and 2005

2007-22
Zlatko Zlatev (UT)
Goal-oriented design of value and process models from patterns

2007-23
Peter Barna (TUE)
Specification of Application Logic in Web Information Systems

2007-24
Georgina Ramírez Camps (CWI)
Structural Features in XML Retrieval

2007-25
Joost Schalken (VU)
Empirical Investigations in Software Process Improvement

====
2008
====

2008-01
Katalin Boer-Sorbán (EUR)
Agent-Based Simulation of Financial Markets: A modular, continuous-time approach

2008-02
Alexei Sharpanskykh (VU)
On Computer-Aided Methods for Modeling and Analysis of Organizations

2008-03
Vera Hollink (UVA)
Optimizing hierarchical menus: a usage-based approach

2008-04
Ander de Keijzer (UT)
Management of Uncertain Data - towards unattended integration

2008-05
Bela Mutschler (UT)
Modeling and simulating causal dependencies on process-aware information systems from a cost perspective

2008-06
Arjen Hommersom (RUN)
On the Application of Formal Methods to Clinical Guidelines, an Artificial Intelligence Perspective

2008-07
Peter van Rosmalen (OU)
Supporting the tutor in the design and support of adaptive e-learning

2008-08
Janneke Bolt (UU)
Bayesian Networks: Aspects of Approximate Inference

2008-09
Christof van Nimwegen (UU)
The paradox of the guided user: assistance can be counter-effective

2008-10
Wauter Bosma (UT)
Discourse oriented summarization

2008-11
Vera Kartseva (VU)
Designing Controls for Network Organizations: A Value-Based Approach

2008-12
Jozsef Farkas (RUN)
A Semiotically Oriented Cognitive Model of Knowledge Representation

2008-13
Caterina Carraciolo (UVA)
Topic Driven Access to Scientific Handbooks

2008-14
Arthur van Bunningen (UT)
Context-Aware Querying; Better Answers with Less Effort

2008-15
Martijn van Otterlo (UT)
The Logic of Adaptive Behavior: Knowledge Representation and Algorithms for the Markov Decision Process Framework in First-Order Domains.

2008-16
Henriette van Vugt (VU)
Embodied agents from a user's perspective

2008-17
Martin Op 't Land (TUD)
Applying Architecture and Ontology to the Splitting and Allying of Enterprises

2008-18
Guido de Croon (UM)
Adaptive Active Vision

2008-19
Henning Rode (UT)
From Document to Entity Retrieval: Improving Precision and Performance of Focused Text Search

2008-20
Rex Arendsen (UVA)
Geen bericht, goed bericht. Een onderzoek naar de effecten van de introductie van elektronisch berichtenverkeer met de overheid op de administratieve lasten van bedrijven.

2008-21
Krisztian Balog (UVA)
People Search in the Enterprise

2008-22
Henk Koning (UU)
Communication of IT-Architecture

2008-23
Stefan Visscher (UU)
Bayesian network models for the management of ventilator-associated pneumonia