Applying the V Model and Axiomatic Design in the Domain of IT Architecture Practice

ARTICLE · JANUARY 2015

DOI: 10.1016/j.procir.2015.07.035

3 AUTHORS, INCLUDING:

René Ronald Bakker

Hogeschool Arnhem and Nijmegen

61 PUBLICATIONS **230** CITATIONS

SEE PROFILE

Stef Joosten

Open Universiteit Nederland

50 PUBLICATIONS **290** CITATIONS

SEE PROFILE

Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 34 (2015) 262 – 267

www.elsevier.com/locate/procedia

9th International Conference on Axiomatic Design – ICAD 2015

Applying the V Model and Axiomatic Design in the Domain of IT Architecture Practice

D. Tarenskeen [a,]*, R. Bakker [a], S. Joosten[b]

[a] *HAN University of Applied Sciences, Arnhem, The Netherlands*
[b] *Open University, Heerlen, The Netherlands*

* Corresponding author. *E-mail address:*debbie.tarenskeen@han.nl

Abstract

This paper applies and discusses the principles of Axiomatic Design for changing IT architecture in health care. It presents three case studies positioned in the field of Enterprise architecture that explore how IT architects, as professionals, manage change and re-design the structure of the IT systems in line with strategic goals. The research approach was to use a light modelling tool, Ampersand, for modelling the Enterprise architecture. Two types of models stand out: Type 1 Strategic IT models in which higher strategic goals are related to requirements for applications and Type 2 Technical management of systems models in which technical risks and risk of system failure in the current IT infrastructure were modelled. To bridge the views of different IT experts in the organization this work uses the customer domain, the functional domain and the physical domain from Axiomatic Design in an extended example in the paper. The V Model is used to bridge the models, and then it is extended with Axiomatic Design principles.

Peer-review under responsibility of the organizing committee of 9th International Conference on Axiomatic Design

Keywords: IT Architecture practice; Enterprise Architecture; V Model; Axiomatic design

1. Context and original research questions

1.1. Research question and case selection

Our original research question was: How can we improve the practice of the IT architect in health care with IT architecture models? First, three exploratory case studies were planned to provide one or more hypotheses about the models of IT architecture. A case study is seen as a mature research method for building theories in this field [1-3]. The cases are examples of paradigmatic cases, because they provide typical information about the IT architecture practice, and extreme cases, because the context is one in which IT architecture is new [4].

In the cases IT architects explore the IT application landscape in order to decide which parts of the landscape have to be changed or replaced. This is a real-life goal for IT architects, as an existing IT landscape always exists. In the cases, changes in the IT architecture were needed because of changing strategic goals of the organisations. In all cases, the starting point has been the document of strategic goals of the health care organisation. The models had to take these goals into account while applying Enterprise architecture scope on the IT architectures [3].

The original idea was to start from research on IT architecture and methods that are widely used such as TOGAF and Archimate, methods for developing and detailing Enterprise architectures. We planned to use models from these methods that would then be input for discussions with stakeholders [3, 5, 6]. However, this is not how the case studies were conducted. The IT architects involved had their own way of modelling and strived to get a grip on the complexity of the existing application landscape. Our research followed the systematic approach that IT experts used.

Although concepts and relations have a similarity to concepts in TOGAF and Archimate, they were not explicitly applied in these models. The IT architects used a proprietary reference model for hospitals based on Weil and Ross [7] and the in the Netherlands broadly accepted ‘Reference domain model for Hospitals’ of Nictiz 2012-2014 [8].

Peer-review under responsibility of the organizing committee of 9th International Conference on Axiomatic Design
doi:10.1016/j.procir.2015.07.035

1.2. Modelling tool

We have used a modelling approach called Ampersand, because of its lean model and its possibilities to formulate rules based on relation algebra for checking consistency of data. Ampersand uses concepts, relations and business rules in a model. These terms in the model are defined in the Ampersand metamodel. Ampersand includes a tool for checking the consistency of the IT architecture data with the formulated business rules. The flexible modelling aspect was especially needed because the models changed often during discussions. The business rules helped to analyse the underlying goals [9, 10].

1.3. Data collected

The resulting data consists of models in different versions, reports and mails with discussions between IT architects, IT senior experts, an Information architect and one of the researchers. After each case an evaluation session of the project and its concrete benefits for the organisation was held.

In case 1: information is based upon models with input from an IT architect and an IT project manager. We had access to reports of 11 theme groups of medical specialists and hospital employees in which requirements where discussed. We had access to drafts for the IT architecture of the future hospital and the technical infrastructure. We had access to end reports of the preparation phase and to a specific plan for adapting a selected application, based on the elicited requirements. Models have been filled with data and the rule engine of Ampersand has been applied for formally testing rules.

In case 2: Information is based on models with input from an IT architect and a senior IT expert from the IT department. Models have been filled with data and the Ampersand rule engine is applied for formally testing rules. We had access to documents describing the strategic goals of the organisation. We developed an instrument for assessing technical risks for the existing applications in cooperation with both IT experts. We had access to an advice of the IT department for adding a data service bus to the architecture for accessing distributed data. During this case, interviews were held with relevant IT experts in the field of technical management of systems.

In case 3: Different versions of the model are discussed with two IT architects in the hospital. The model is currently under construction. Models have been filled with data and the Ampersand rule engine has been applied for formally testing rules. We have access to the drafts and definitive goal architecture document and drafts for the IT architecture diagrams.

2. Axiomatic Design

2.1. Overview of Axiomatic Design Theory

The ideas of Axiomatic Design (AD) originate from industrial production and industrial systems, but they are relevant for software and hardware systems. The design method bridges different domains when describing the design of a suitable system. Two axioms stand at the basis of this method [11, 12]. It can be mathematically demonstrated that designs following these axioms are better designs, in the sense of less complexity and less costs [13, 14].

AD assumes that designing systems requires input from different domains. The domains that are introduced are: the customer domain, where customer needs are elicited, the functional domain, in which functional requirements are positioned, the physical domain for describing the design parameters of the system and the process domain that complements the foregoing domains with information for the process of manufacturing the system.

The first axiom states that when a design is made all Functional Requirements (FRs) must be formulated on a fundamental level, in such a way that they are independent from each other. The FRs are later mapped to its corresponding Design Parameters (DPs). A DP describes a property or characteristic of the system. It is expected to fulfil the related FR.

The second axiom states that for every DP we should minimize the information content [11]. It states that if we can choose between alternatives for a DP, choosing the DP with the highest probability of success leads to the best design. As low information content minimizes complexity, and thus contributes to a better chance of implementing the DP. Theoretically the information function is defined with a probability function. After that the probability of fulfilling the requirement (FR) with the realized DP is estimated. In software engineering we found in accordance with the comment of Suh that often there exists no observable Design Range where a DP fulfils the FR, because the DP either fulfils the FR or it does not [11, p247].

Thus, it is not possible to estimate the probability of fulfilling the FR with the DP. But an estimation of the probability of delivering the DP can be made. One can for instance compare difficulties in realizing the DPs with different software constructing processes and tools. The difficulty then depends on the needed knowledge for realising the DP. More often than not, knowledge of different tools and libraries is necessary. This increases the information content of the DP. In practice one often refers to examples of DPs as “proven technology”, when these are demonstrable and operational in other organisations, and thus give an indication of its probability of successful realization. In all case studies definitive decisions about DPs were made based on “proven technology”. This was not explicitly examined, but was found in the cases.

2.2. Axiomatic Design in this study

The case studies were exploratory and resulted in different models that conformed to the requirements of the designers. However, the models were fragmented and each showed different parts of the reality of the different IT experts. We needed a Design Matrix (DM) to combine the models, and AD for assessing the quality of the models.

We have first applied the V Model in this context to map the business requirements to system functionalities for the

existing IT application landscapes. The V Model is explained in section 4.

We then transformed the table in the V Model to a DM, we replaced business requirements with FRs and system functionalities with DPs. Then we placed the DM at the bottom of the V Model for communicating it to the IT architects and show the connection to the existing IT landscape. We expect to improve the design with AD.

3. Cases

3.1. Three case studies

The case organisations were all in the process of reorganizing the core business, the care process, due to changes in governmental regulations, changing vision on health care and a more central role of the patient in the interaction with medical personnel. They have an IT infrastructure in operation that functions sufficiently for the current requirements. The board of directors in all three cases wanted to re-evaluate the IT systems in the light of new requirements.

The researchers collaborated with the IT architects or with the IT department. The models were developed iteratively. Models that "were interesting" led to more versions, sometimes up to 10 versions. Models that had no relevance to the practice of the architects were abandoned after one or two versions. The rules for checking consistency of models were formulated in natural language by the IT architects and formalized by the researchers.

Table 1 summarizes similarities and differences in the context of the case studies.

Table 1. Context in different case studies.

Characteristics	Case 1	Case 2	Case 3
Changing Strategic vision	x	x	x
Evaluation of IT infrastructure	x	x	x
Need for new design of IT infrastructure	x		
Need for replacing parts of the IT infrastructure		x	
Need for extension of IT infrastructure			x
Contact IT architects	x		x
Contact IT experts from IT departments		x	x

3.2. Health care organisations in case studies

3.2.1. Case study 1: National hospital for child oncology

The reason for developing the architecture was the establishment of a new national hospital for child oncology, that would integrate laboratory functions and treatment, integrate knowledge from research in this area and treatment and coordinate care centres across regions in the Netherlands. Information technology that existed in an academic hospital nearby would be reused where possible. The IT architecture model, that was eventually selected by the IT architects can be seen as a simplified version of an enterprise architecture in which strategic goals were detailed in requirements for applications.

3.2.2. Case study 2: Medium sized regional centre in health care

The case organisation in health care functioned mainly as a care organisation and provided home care and care for the elderly. It had 5000 employees and about 3000 volunteers at the time of study. The IT architect and IT expert started with formulating strategic goals, because the organisation was in the process of changing its health care vision and strategy. The ultimate goal however, was to judge if the current system was up to the required changes and could be maintained in its current form. The board of directors was planning to update and replace parts of the system that were not functioning according to state of the art requirements.

3.2.3. Case study 3: Ongoing case: medium sized hospital

In the third case study a medium sized hospital in the Netherlands is being studied. The hospital is in the last stage of changing its paper-based operations to digital operations in the health care processes. The goal of the IT architects is to deliver a consistent vision of the changes in health care services as foreseen and the specified supporting IT functionality.

4. Findings

The first two case studies resulted in five distinct architecture models, these have been iteratively developed in a total of 30 versions. Figures 1 and 2 show examples of the two types. We describe the two types in the next section.

4.1. Types of IT Architecture models

The five architecture models can be ordered in 2 types of models: one type in which strategic goals are leading and one type in which the technical management of system components was a priority. It makes a difference whether the IT experts apply an enterprise level point of view or a technical management point of view. Analysis of the rationale of involved parties point out that underlying goals for constructing the architecture models differ. These goals determine the ordering and structure of concepts. The difference is so fundamental that connecting different models is difficult or leads to a meaningless mapping between the elements of different models. Due to the complexity and chaotic structure of the relations between the models, checking whether technical systems sufficiently support strategic business goals is not possible with these models.

Type 1 Strategic IT models show models in which higher strategic goals are related to requirements for applications and decisions about the IT architecture. See example in Figure 1. The architects wanted an overview the information that would abstract from the complexity numerous elements and relations between them. The goal of the model was to present a view of "Fit and Gap" of existing applications. Business rules in Ampersand are applied for signalling gaps.

The mapping of requirements to applications proved to be a bottleneck because the information about the internal structure of applications was excessive and detailed. The project was cancelled because of a decision that in the first

phase of implementation all existing IT infrastructure of a nearby academic hospital would be used “as is”. Adaptations for the IT architecture were delayed until 2015.

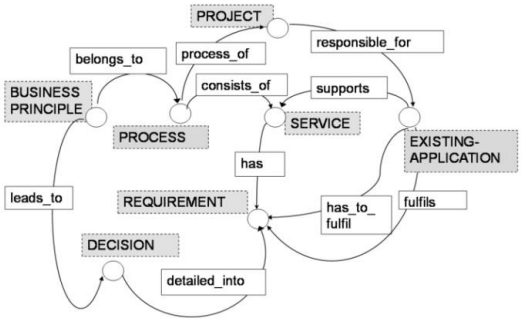

Figure 1. Example of model of type 1 Strategic IT

Type 2 Technical management of systems models consist of 3 models in which technical risks and risk of system failure in the current IT infrastructure were modelled. In terms of AD theory, Type 1 models show the customer needs and the functional requirements that are derived from them. Type 2 models represent an image of the runtime system with system characteristics and dependencies.

For type 2 models the researchers developed an instrument to assess the risks of certain applications. The instrument is based on theory and adapted in interviews with the IT experts of the organisation. It was based on Andreou [15], NEN-ISO Standard Riskmanagement [16] and Lock and Sommerville [17]. This instrument was tested in interviews with 2 other IT experts from the IT department.

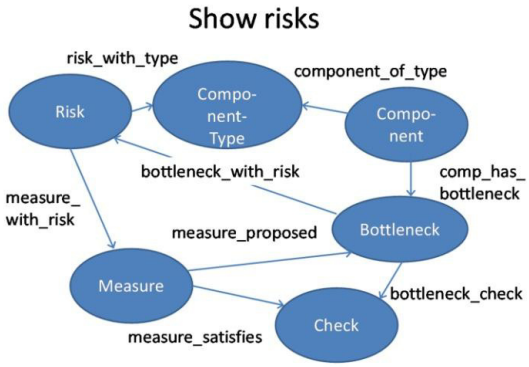

Figure 2 Example of model of type 2 Technical management

A remarkable finding in this case study was that after the technical models were constructed, relations to the strategic goals of the organisation were not added to the models. Showing that these IT experts did not include the strategic goals in their models, and had no reference to them from the models that they considered meaningful, hereby confirming that the customer domain was not relevant in their models.

4.2. Matrix in V Model

In the context of case 3 we expected a multitude of relations from strategic goals to IT systems that would be difficult to interpret. Therefore the researchers developed a model in which the business functions and IT system functionality would be defined on a detailed level. The details were used to demonstrate the possibility to bridge the two types of models. We had not yet applied AD in this stage of research.

We constructed a V Model that showed the relation between two different points of view in IT architecture on a deeper level of detail. We used zigzagging between domains to get to this level. This V Model can be seen in figure 3. The specification of strategic goals to sub goals and requirements stemming from business can be found on the left side of the V. The right side of the V is used for mapping system functionalities upwards to a technological IT architecture. The latter is a representation of the runtime systems that are in use in the organisation. When comparing the V Model with the Design Method of AD the left leg shows the Customer needs mapped to the FRs in the functional domain. Since IT architects work with these two domains, they can model them in one line.

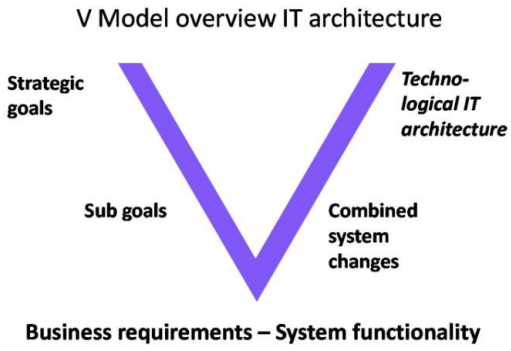

Figure 3. V Model for overview of IT Architecture.

The right leg shows the mapping of different levels in the physical domain, where DPs on lower levels are mapped onto DPs in higher levels on the right leg. This line does not combine two domains in the AD. It shows only the running/runtime system and the required changes. We have used this diagram for communication purposes with IT architects and found a natural understanding. Even though this V Model was grounded on a deeper level on zigzagging from functional domain to the physical domain. The process domain is out of scope, because designing of program coding is not part of the models. The runtime system does not consists of program modules but of different system components that can be started separately and communicate during runtime.

An illustration of the matrix at the bottom of the V Model in case 3 can be found in table 3, without AD. In Table 2 four sub goals are shown. These sub goals belong to the strategic goal of increasing regional collaboration.

Table 2. Sub goals left side V Model

	Sub goals of Increasing regional collaboration
1	Improve ways for requesting medical examination by first-or primary care
2	More and better ways to judge examination results
3	Better and more direct contact between first-or primary care and hospital
4	Permissions for first-or primary care to access EHR Electronic Health Records or EMR Electronic Medical Records of their own patients in hospital EMR

In Table 3 the matrix at the bottom of the V is shown. In this table for two out of the four sub goals the business requirements are formulated and determined. They are linked one by one to required system functionality. AD is not yet applied here.

Table 3. Matrix V Model

	Business requirement	System functionality
1.1	Electronic request for lab examination by GP (general practitioner)	Access and interoperability for electronic requests for lab system hospital
1.2	Electronic request for lab examination by GP	Processing electronic requests by lab system
1.3	Electronic request for radiology examination by GP	Access and interoperability for electronic requests for radiology system hospital
1.4	Electronic request for radiology examination by GP	Processing electronic requests by radiology system
1.5	Electronic request for medical examination by GP	Access and interoperability for electronic requests for medical examination system hospital
1.6	Electronic request for medical examination by GP	Processing electronic requests by medical examination system
1.7	Service catalogues requests by first-or primary care	Access and views for electronic requests for service catalogues
2.1	Electronic consultation or automatic reception of images in required quality	Access and views for Image processing system PACS2 hospital
2.2	Electronic consultation or automatic reception of results of examinations by specialists of requested examinations	Access and views for requesting results lab
2.3	Electronic consultation or automatic reception of results of examinations by specialists of requested examinations	Access and views for requesting results radiology
2.4	Electronic consultation or automatic reception of results of examinations by specialists of requested examinations	Access and views for requesting results medical examination

4.3. Example of Axiomatic Design

We rewrite the matrix business requirements 1.1 and 1.2 in the form of a DM in the meaning of AD, by setting a one on one mapping from FRs to DPs and claim that the DPs can be fixed, in such a way that DPs that are on the left of other DPs are not changed.

In the example in table 4 is seen that one of the FRs, FR3, is referencing 3 DPs. The FRs are:

1. Access electronic request in lab system
2. Process electronic request in lab system
3. Handle electronic request for lab examination by General practitioners

They can be combined together in the following way with DPs. The DPs are:

1. Security system for electronic requests for lab system hospital
2. API Application Programming interface for electronic requests by lab system
3. Service for lab requests General practitioners

The first two DPs will be realized in the backend of the server program, both DPs are used by the third DP that combines them to provide the service needed by the user for handling the electronic request.

Table 4. Example Design matrix case 3.

	FR	DP1	DP2	DP3
1.	Access electronic request in lab system	x		
2.	Process electronic request in lab system		x	
3.	Handle electronic request for lab examination by GP (general practitioner)	x	x	x

This can only be a responsible design according to AD if the first two DPs are “fixed” [11]. In software the fixation is realized by encapsulation of the functionality that can be called through a specific, defined and fixed “interface” or definite system function call. Underlying functionality can then be changed without affecting the calling system functions as long as the “interface” is not changed. This practice is common practice in Software engineering and is regarded as good design [18].

By fixing the underlying DPs numbers DP1 and DP2, the design is triangular and decoupled.

5. Discussion

5.1. Goal oriented character of models

We presume that underlying goals of IT experts are the reason for different structure and ordering of models, in the cases.

5.1.1. Strategic IT models of IT architects

The IT architects have a task in making sure that IT is up to date and can support the organisation as a whole, the health care processes and its supporting processes. They make decisions about the structure of the IT application infrastructure and perform a fit and gap analysis. They advise which parts have to be changed or adapted for needed functionality. The traditional idea of Business IT alignment is the responsibility of the IT architects [19].

5.1.2. Technical management models of IT operations and IT departments

The second group of IT experts work in the IT department and manages the IT systems.

The IT departments have a responsibility for the totality of the IT infrastructure. They guarantee to operate a running, stable, and secure system. They adapt the applications when a detailed request with permissions from higher management is

received. They make decisions about rules for users and constraints for adaptation in order to guarantee stability and security. The IT department has the most extended knowledge of applications, their functionality and the options for storage and adaptations that are activated in case of system failure. According to Bass et al. software architecture contributes to the quality attributes of software [20].

5.2. Bridging models with Axiomatic Design

We have applied the V Model in this context to map the business requirements to system functionalities for the existing IT application landscapes. We later used the principles of AD to replace business requirements with FRs and system functionalities to DPs. We have placed the DM at the bottom of the V Model for communicating it to the IT architects and show the connection to the existing IT landscape.

The DPs are designed in such a way that ideally every FR can be satisfied with one or more DPs in an independent way. Our goal to show traceability in design can thus be achieved.

Another advantage of applying AD for design is the opportunity to change FRs without compromising other functions, because in AD the FRs are independent of each other. A different approach, Model driven engineering, has been taken by Verelst in his research on Normalized systems, where software systems are generated in such a way that every system function is independent of other system functions [21]. This approach is not suitable in our cases because a replacement of the infrastructure as a whole is not feasible in this complex, operative context.

6. Conclusions

We have found two types of models of IT architecture that both contain relevant information for evaluating the IT infrastructure in supporting the goals of the organisation. Bridging the models is necessary, because IT architects in the study need to relate their strategic goals to the IT systems in order to check traceability and soundness of design of the IT architecture. The gap can be overcome by using the DM of Axiomatic Design. For visualizing the required changes to the existing IT architecture the V Model can be used. Axiom 1 is compatible with best practices in IT development. Axiom 2 needs further research.

7. Future research

Our research will proceed with the study of bridging different IT Architecture models in an organisation with the V Model and AD. Priority will be given to the study of effects of applying AD on the practice of IT architecture.

Acknowledgements

The research is part of PhD research of D.Tarenskeen and was performed in the HAN University of Applied Sciences, Arnhem, The Netherlands and the Open University, Heerlen, The Netherlands.

References

[1] Eisenhardt KM. Building theories from case study research. Acad Manage Rev. 1989; 14(4):532-550.
[2] Yin RK. Case study research: Design and methods. Fourth edition. Sage publications; 2009.
[3] Lankhorst M. Enterprise architecture at work: Modelling, communication and analysis. In. Springer 2013; p. 269-302.
[4] Flyvbjerg B. Five misunderstandings about case-study research. Qual Inq. 2006; 12(2):219-245.
[5] The Open Group. TOGAF Version 9.1 Evaluation copy. The Open Group; 2011.
[6] The Open Group. Archimate 2.0 Specification. Zaltbommel: Van Haren Publishing; 2012.
[7] Ross JW, Weill P, Robertson D. Enterprise architecture as strategy: Creating a foundation for business execution. Harvard Business Press; 2006.
[8] Fischer M, Smeele F. Referentiedomeinenmodel ziekenhuizen versie 2.2. 2014; 2015. Available from: www.nictiz.nl/module/360/657/12001A_Referentiedomeinenmodel_ziekenhuizen_versie_2_01.pdf.
[9] Michels G, Joosten S, van der Woude J, Joosten S. Ampersand. In: Relational and Algebraic Methods in Computer Science. Springer 2011; p. 280-293.
[10] Michels G, Joosten S, Woude Jvd, Joosten S. Ampersand applying relation algebra in practice. In Proceedings of the 12th international conference on Relational and algebraic methods in computer science. Rotterdam, The Netherlands: Springer-Verlag 2011; 280-293.
[11] Suh NP. Axiomatic Design: Advances and Applications (The Oxford Series on Advanced Manufacturing). New York Oxford: Oxford University Press; 2001.
[12] Kulak O, Cebi S, Kahraman C. Applications of axiomatic design principles: A literature review. Expert Syst Appl. 2010; 37(9):6705-6717.
[13] Malaek SMB, Mollajan A, Ghorbani A, Sharahi A. A New Systems Engineering Model Based on the Principles of Axiomatic Design. J Ind Int Inf. 2015; 3(2).
[14] Togay C, Dogru AH, Tanik JU. Systematic component-oriented development with axiomatic design. J Syst Softw. 2008; 81(11):1803-1815.
[15] Andreou AS, Tziakouris M. A quality framework for developing and evaluating original software components. Inf Softw Technol. 2007; 49(2):122-141.
[16] International Electrotechnical Commission. NEN-ISO/IEC 31010:2009. Riskmanagement - Risk assessment techniques. Geneva, Switzerland: IEC; 2009.
[17] Lock R, Sommerville I. Modelling and analysis of socio-technical system of systems. In Engineering of Complex Computer Systems (ICECCS), 2010 15th IEEE International Conference on. IEEE 2010; 224-232.
[18] Larman C. Applying UML and patterns: an introduction to object-oriented analysis and design and iterative development. Third Edition. Upper Saddle River. NJ: Prentice Hall PTR; 2005.
[19] Henderson JC, Venkatraman N. Strategic alignment: leveraging information technology for transforming organizations. IBM Syst J. 1993; 32(1):4-16.
[20] Bass L, Clements P, Kazman R. Software architecture in practice. Third Edition. Upper Saddle River, NJ: Addison-Wesley Professional; 2012.
[21] Verelst J. The influence of the level of abstraction on the evolvability of conceptual models of information systems. Empir Softw Eng. Oct 2005; 10(4):467-494