Towards Scenario-based Discovery of Domain-Specific Patterns: a Case Study

Philips Huysmans

Antwerp Management School
Antwerp, Belgium
Email: philip.huysmans@ams.ac.be

Jan Verelst and Gilles Oorts

Normalized Systems Institute
University of Antwerp, Belgium
Email: {jan.verelst,gilles.oorts}@uantwerp.be

***Abstract*—The lack and need of prescriptive design knowledge in enterprise architectures is well documented. While knowledge of various disciplines that are part of enterprise architectures is captured in principles or patterns, no integration of this knowledge is available. In order to work towards such knowledge documentation, we propose an inductive documentation of domain-specific patterns. These patterns can be observed by analyzing different design alternatives, and evaluating them against qualitative criteria, such as evolvability. In this paper, we present a method to systematically analyze and document design alternatives in a domain, building on scenario-based architecture evaluation methods. A case study is presented in which the proposed method is applied. Based on the findings of this method, domain-specific enterprise architecture patterns can be proposed in future research.**

Keywords–Modularity; Patterns; Design Structure Matrices.

I. Introduction

It has been argued that the wealth of nations relates with their ability to deal with economic complexity [1]. In this perspective, the best performing countries are not the countries with the highest qualities of inputs, but those which use the recombinational potential of already available inputs to create more diverse and complex products. Growth predictions are based on the ability to create different outputs by adding a few inputs to the current production capabilities, rather than the more classical focus of measuring how much value is added to raw materials or intermediate products. As an example, agricultural efforts in a developed versus a developing country can be differentiated based on the ability to integrate with a logistics network, a supply network, a knowledge network, a financial network, etc.

This promise of exponential growth by leveraging the recombination potential is well described by combinatorics theory. In practice however, the drawbacks of combinatorics are more easily observable than its advantages. Changes to any artefact result in ripple effects, causing more changes than anticipated. As a result, change becomes complex and expensive. In dynamic markets, change requirements occur at a frequency which prevents organizations to consider change as an adaption of a steady state, but necessitates the application of changes at a constant pace. As a result, products and services, and their combinations, increase in complexity, which again limits the possibility to reap the benefits of the recombination potential because of the ripple effects.

Prescriptive solutions which prevent these ripple effects are available in various disciplines. The idea of applying patterns to codify design knowledge is widespread in software architecture. In business process modeling, modularization patterns are described by e.g., [2]. On the management level, modularity and coupling are studied as well [3], and certain patterns are described there as well [4]. In practice however, organizations have to design artifacts in each of these disciplines (i.e., an organizational structure performing certain processes which are supported by software systems). Put differently, the design knowledge of these different disciplines needs to be combined and integrated. The field of enterprise architecture has these disciplines in scope. However, the lack of deterministic design in each of these separate disciplines demonstrates the difficulty and complexity of performing such a design in an integrated way. Unsurprisingly, few patterns are known in the field of enterprise architecture.

Rather than attempting to solve the integration of design knowledge of different disciplines in general, we believe a more feasible approach is to start with the documentation of domain-specific patterns. A domain has its key challenges, similar artifacts, and similar integration issues. This limitation of scope can make the documentation of design knowledge more focused, and hence, more feasible in the short term.

In this paper, we therefore present a method which was used to systematically research couplings between artifacts on the organizational, process and Information Technology (IT) level of different organizations in a certain domain. This method is based on scenario-based analysis methods. These methods propose to compare different architectures by evaluating how well they support certain scenarios. By documenting relevant domain-specific changes as scenarios, we can systematically research which designs are susceptible to ripple effects in various change scenarios. We use design structure matrices to document the modular couplings (which cause ripple effects) between artifacts. Different design alternatives can then be documented and, if sufficient scenarios are tested, be proposed as design patterns for that specific domain.

In Section II, we introduce the building blocks of the method. In Section III, we present the designed method. Section IV demonstrates the method by applying it to three different organizations in the hospital sector. Finally, we discuss our findings in Section V.

II. Method Building Blocks

A. Scenario-based methods

In order to compare and gain insight in different architectural solutions, a scenario-based approach for decision making can be adopted [5]. The Software Architecture Analysis Method (SAAM) enables the usage of scenarios on a software level [6]. Various approaches have already elaborated

 ISBN: 978-1-61208-534-0

on SAAM, such as the Architecture Tradeoff Analysis Method (ATAM) [7] and the Architecture-Level Modifiability Analysis (ALMA) [8]. SAAM is the simplest of the software evaluation methods. While various methods extended SAAM with other elements, these additions clearly focused on the evaluation of software architectures. The basic structure of SAAM is sufficient for our approach. SAAM itself enables the expression of different quality claims of software architectures such as, amongst others, modifiability, exibility, and maintainability. The realization of these quality claims in a certain software architecture is then evaluated using scenarios. SAAM consists of six main steps, which are generally preceded by an overview of the business context and the functional requirements of the system.

We are not the first to adopt SAAM in a context that is different from software. For example, in the paper "Characterization of Enterprise Architecture Quality Attributes" [9], the authors clearly state the use of the work of Bass et al [10] regarding software architectures, software quality attributes and scenarios as a basis. Moreover, it has been argued that scenario-based methods can be applied in any field where modifiability is a concern [11][12].

B. Enterprise Architecture

Enterprise architectures present an overview of strategic goals and organizational and technical artifacts of an organization, in order to manage the challenges of change and complexity. Enterprise architects mainly aim to reduce the complexity by creating abstractions from real-world artifacts by creating models [13]. These models are grouped in architectural levels or layers. Different enterprise architecture frameworks propose different layers, or require that organizations define their own sets of layers [14][15][16]. It has been argued that most publications on enterprise architectures report on contributions which can be located on a single layer, while few authors address integrating multiple layers [17]. A modeling approach for documenting coupling across different layers is usually not proposed in the various frameworks. As such, a complementary documentation model for these cross-layer couplings needs to be adopted.

C. Design Structure Matrices

The modularity paradigm provides tools and models which allow an explicit focus on modular dependencies. Recently, organizational modularity has gained much attention in research and practice [3]. In this paradigm, it is argued that product, processes and organizational structures can be regarded as modular structures. Moreover, certain authors claim that modularization on, for example, the product level drives modularization on other levels as well. This is referred to as the mirroring hypothesis [18]. While we do not explicitly use this hypothesis, it indicates how modularity can be used as a way to analyze the integration of different architectural layers. By adhering to the modularity paradigm, we can use theories and tools which apply modularity in our proposed method.

More specifically, we will adopt Design Structure Matrices (DSM), which were heavily used by Baldwin and Clark. DSMs provide an accepted and well-defined notation to represent architectural components and interfaces [19][20]. They are used in traditional modularity approaches (e.g., product modularity) to visualize dependencies between and within modules. A modular dependency occurs when a change to an aspect of a module could require changes to other aspects, within that module or in other modules.

III. Resulting Method

The six steps of SAAM will be used as the general outline of our method. The first step (i.e., develop scenarios) is identical to the original method, with the exception of the different nature of the selected scenarios (i.e., on the enterprise architecture level instead of on the software level). A scenario can be viewed as a brief description of a stakeholder's interaction with the system [6].

For the second step (i.e., describe the architecture), we propose the use of a design structure matrix (DSM). The different architectural layers (e.g., organizational, process and IT layers) can be conceptualized as different modular structures, and coupling between modules of different layers can be documented as modular dependencies.

In the third step, SAAM advises to classify and prioritize the scenarios. For each scenario, it needs to be determined on which layer of the DSM (as constructed in step 2) it requires a direct functional change. The scenario will then be positioned as a design element in one of the modules. For example, a scenario indicating a technological change should be placed in the IT module. In contrast, a scenario indicating a reorganization should be placed on the organization module.

In the fourth step, SAAM advises to individually evaluate the indirect scenarios. However, ripple effects can be present in direct scenarios as well. The presence of a ripple effect in a direct scenario would mean that while the architecture supports the scenario in its current form, it could require increasing adaptations once the organization needs to scale. Therefore, an architecture which does not contain ripple effects for direct scenarios will be preferable to an architecture which does contain ripple effects for direct scenarios. Consequently, we advise to include an evaluation of the direct scenarios as well. For each of the scenarios, any design parameter which will be affected by the implementation of the scenario needs to be considered. These design parameters are then added to the DSM. It should be noted that these design parameters can be positioned on other layers than the original scenario. This step already creates awareness for the analyst to take all organizational aspects into account when evaluating scenarios. An "x" should be added in the intersection of the column of the scenario and the row of the design parameters that this scenario affects.

In the fifth step, SAAM advises to assess the scenario interaction. In our approach, this requires the completion of the DSM. For every intersection, a possible dependency needs to be evaluated. Newly found dependencies should be indicated with an "x". This allows for a detailed and systematic evaluation of interactions between previously unknown scenarios or design parameters. However, the DSM can become too complex to be used as a basis to communicate. Especially the identification of chained dependencies can become complicated.

In the sixth step, SAAM advises to perform an overall evaluation. Using the dependency chains identified in step five, insight in architectural issues can be communicated easily to involved stakeholders. The developed artifact can contribute to a systematic approach to identify, communicate, and create awareness concerning design choices. A comparison of different design alternatives can create pattern candidates, which can be further evaluated qualitatively.

IV. CASE STUDY: APPLICATION TO THE HOSPITAL DOMAIN

For this demonstration, the hospital sector was selected. The selection was motivated by the dynamic nature of the sector, and the similarity of organizational size of the prominent players. Large variations in size could have an impact on preference for certain architectural characteristics. Overall, three cases were conducted, which consisted of at least two in-depth interviews and additional review questions through email. The case participants were selected based on business experience and knowledge regarding the high-level IT architecture.

1) Step 1: Identify scenarios: The first step was performed by organizing brainstorm sessions. After an initial draft of the scenarios, their relevance was checked by discussing them with stakeholders from the other cases. The respondents agreed that the resulting set of scenarios either (1) were likely to occur in the near future, or (2) had an important impact on their organization in the past.

- Scenario 1: **Changing risiv code**: the risiv code is an identification number for a governmental entity related to sick leave and invalidity insurance. Each investigation or procedure performed in a hospital needs to append such an identification number to determine the reimbursement level of medical costs to the patient. Changes in legislation can change which code needs to be attributed to a certain procedure, or can change the coding scheme as a whole.
- Scenario 2: **New medical cabinet supplier**: In most hospitals, a decentralized supply of medicines is used. The medical cabinets are managed using an IT system which is integrated with the purchasing system. Moreover, the medicine usage of every patient is registered and charged individually. Consequently, no medicine may be retrieved without patient identification.
- Scenario 3: **Introduction of a new medical specialization**: Especially in academic hospitals, new research can result in improved methods or even new specializations. In order to support these activities, integration with existing systems and procedures need to be constructed, as well as new artifacts specific to the new medical activities.
- Scenario 4: **Changes in the patient registration process**: During emergencies, regular registration or consultation, patients need to register before being treated. A file is kept for each patient to be able to consult previous procedures or treatments. During registration, data from identification cards (regular id or medical id) needs to be extracted.
- Scenario 5: **Changes in the patient classification system**: Patients are classified for various purposes. In many hospitals, the type of registration impacts the invoicing and reimbursement procedures.
- Scenario 6: **Changes in the procedure classification system**: In most hospitals, a wide variety of clinical procedures (1000+) can be performed. The classification code for a procedure is used during communication with, for example, the sterilization department, which prepares the correct set of tools and delivers them to the operation room. However, this classification is also used in other contexts, such as communication in professional journals, which uses a possibly different and international classification scheme. Especially in academic hospitals, much discussion regarding the selection of a certain classification system are reported.
- Scenario 7: **Opening a new site**: The final scenario attempts to reflect on the scalability of the current architectures. While no functional changes to existing systems are required, duplication of existing systems, information and positions greatly increase the complexity of the organization as a whole. Nevertheless, the current mergers and push towards centralization in the sector resulted in an agreement on the importance of this scenario by all participants.

2) Step 2: Describe the architecture: Currently, none of the organizations has a documentation of their architecture. One organization has started an enterprise architecture program based on the lack of flexibility and presence of integration issues. After educating several employees, it was concluded that the required documentation and formalization, combined with the changes which require an effort to keep the models up-to-date, resulted in too much effort. Moreover, management was not convinced of the relevance of the resulting documentation.

All three organizations have two main organizational entities: an administrative and a medical entity. Both entities have separate staff and separate IT systems. A distinction between the organizations can be made based on the academic or general nature of the organization. Moreover, a distinctive characteristic is the mode of employment: medical staff can be directly employed by the hospital, or operate independently. Beyond administrative differences, this distinction impacts the sharing of information and the preference for the selection of software packages.

3) Step 3: Classify and prioritize scenarios: In this step, the scenarios need to be classified in the different architectural layers. Scenarios 2 (new medical cabinet supplier - stakeholder management), 3 (introduction of a new medical specialization - business model), and 7 (opening a new site - value clusters) are strategic in nature and can therefore be positioned in the organizational layer. Scenarios 1 (changing risiv code), 5 (changes in the patient classification system), and 6 (changes in the procedure classification system) reflected mainly organizational changes as well, and are therefore classified as such. Scenario 4 (changes in the patient registration process) is considered as mainly a process change.

When asked to position the scenarios as direct or indirect, our respondents indicated that only the scenario 1 (changing the risiv code) could be considered a direct scenario. For the other changes, changes to the current architecture would be required. The prioritization of scenarios was not elaborated upon, since this would only impact the selection of a pattern. Currently, this research focuses on the identification of modular couplings to motivate the selection of certain design alternatives. The formulation of actual patterns is too ambitious for the current cases.

4) Step 4: Evaluate the scenarios: In this step, the scenarios are evaluated. During this step, the DSM should be filled. An example DSM for this case in presented in Figure 1.

a) Scenario 1: Changing risiv code: Only a single hospital claimed that this scenario could be supported directly, because the risiv codes are linked to the procedures by a centralized invoicing department. The doctors of various departments do not need to be involved with changes or new legal requirements. The other hospitals employed both directly employed and independent doctors. As a result, multiple applications were needed to register billable activities. For certain departments, activities are registered and managed by an invoicing department, while other departments interface directly with the invoicing application. A change in risiv code can therefore affect one, two, or many applications, based on the design alternative employed.

b) Scenario 2: New medical cabinet supplier: The systems supporting medical cabinets from different suppliers use various patient identification codes. In the first case hospital, a different (internal) patient identification code syntax was used, and personnel had to convert the code formats manually. In order to remedy this situation in the future, the hospital will include its own patient identification code syntax as a requirement during cabinet acquisition. A new medical cabinet supplier would then result in the integration of a new external application in the application landscape, but have no impact on the manual (and time-consuming) processes. As such, a design rule for this dependency will be created.

In the second case hospital, a supplier switch was made recently. The design rule for the patient identification code syntax was imposed here as well. Moreover, the external software provided an application programming interface, which allowed integration with the pharmacy order administration and patient administration. A specialized message bus (HL7) was used to make this integration.

In the third case hospital, no experience with this change was present, and no design rules have been formulated to guide a future acquisition process.

c) Scenario 3: Introduction of a new medical specialization: Our respondents indicated that the most impactful change for incorporating a new medical specialization is the development and integration of new software applications. The organizational impacts of adding new processes and assigning locations are well-known. In contrast, previous integration experiences have caused several maintenance issues. In the first and second case hospitals, this has led to the use of a middleware bus (HL7). In the second case hospital, no architectural solution for integrating new applications is present.

d) Scenario 4: Changes in the patient registration process: The increased adoption of electronic IDs has resulted in registration process improvements. However, the effort required to implement these improvements varied across the hospitals.

In the first case hospital, the registration procedure is mainly centralized. Three different registration desks are available, which each handle the same registrations and use the same processes. They are responsible for all registrations.

In contrast, the second case hospital has a combination of centralized and decentralized registration desks. Changes to the procedures followed by registration desks need to be implemented in many different places. An example is the introduction of regional hubs: each hospital will need to integrate with such a hub, so doctors with a therapeutic relationship with the patients have a central repository. Since information from all registration desks will need to be included, every desk is impacted.

In the third case hospital, a combination of centralized and decentralized registration desks is used as well. The resulting complexity has led to a specific organizational role which is created to manage the process of distributing work across registration desks.

e) Scenario 5: Changes in the patient classification system: In the second case hospital, patients are categorized in a classification system during registration. This classification is the input for the invoicing process. Because of evolving structure of the classification structure, a re-ordering effort took place to simplify the structure. However, this initiative was halted since the changes to the invoicing applications proved to be too complex. The third case hospital reported similar issues, and noted that the impacts of changing their classification system would impact additional processes and applications.

In contrast, the first case hospital did not use a patient classification scheme, because the invoicing department bases its processing on the raw data of the procedures performed and medicines used. As such, the invoicing process has less dependencies on derived data.

f) Scenario 6: Changes in the procedure classification system: The third case hospital reports a vast impact of changes in procedure classifications. Soon, a new version of the official classification scheme is expected. This scheme describes the treatments, diagnoses and procedures performed which need to be reported to the government. Currently, a team of 10 employees has the full-time job of determining correct classification codes based on the data of the medical file and the lab results. Changes to the reporting scheme are expected to result in retraining and data changes.

As an example of the impact, we mention the interface between the surgical and sterilization departments. The surgical department needs to communicate its need for sterilization of tools. In the fist case hospital, a classification for the tools is known in the surgical department, and is linked to the procedure classification scheme. In the second case hospital, instead of using tools classification to communicate, the procedure classification scheme is used.

This distinction shows the difference in impact of scenario 6: based on the way of communicating, different departments will be impacted. In general, three possibilities are observed for mapping data between the classification schemes: performed by the surgical department (case 1), in the sterilization department (case 3), or shared on the HL7 bus (case 2). It should be noted that the mapping of data on the HL7 bus introduces business knowledge on the integration bus.

g) scenario 7: Opening a new site: The scenario of opening an extra site allows to reflect of the scalability of the organization as a whole. It does not require new module types, only additional instances of existing ones. Nevertheless, much of the infrastructure is currently not designed to handle such scaling: for example, the data structure of the reference lists of patients of a certain hospital service would have to be redesigned, since only patients from that service on the current site should be included. Moreover, handling of permissions to applications and data would need to include awareness of the different sites. In the third case hospital, this scenario was compared to the future merger with another hospital. Currently, efforts to standardize patient administration processes and employee relations are in progress, in order to bring the merger closer to the scenario of opening a new site. However, the impact on governance structures and organizational culture demonstrate that the impact of these changes is outside the scope of the current approach.

5) Step 5: Assess scenario interaction: In discussion with the respondents, additional dependencies not directly related to scenario interactions were analyzed next, and added to the DSM in Figure 1. This information is crucial to estimate the size of ripple effects, since this documents knowledge which is distributed in the organization. For example, the applications dependent of the syntax of patient identification needed to be gathered from the different application owners, since no centralized knowledge regarding this impact was present. The interviews show that the scheduling application was impacted in case 1, the invoicing application was impacted in all cases, the operation management application was impacted in case 2, and the integration on the HL7 bus was impacted in cases 2 and 3. Moreover, dependencies between scenarios, such as changing the patient classification system and changing the enrollment process can be identified. These dependencies are crucial for detecting chained dependencies. The resulting complexity of the model can be addressed by generating dependency chains that only focus on design parameters relevant during a certain analysis.

6) Step 6: Perform overall evaluation: In order to propose domain-specific patterns based on the (absence of) identified couplings, a comparison of the change impacts in the different designs needs to be made.

As a first observation, the centralization of registration desks increases the flexibility of the first case hospital. Changes in patient registration procedures can be implemented in one or a few desks, without integration issues with other desks.

A second example where centralization benefits the flexibility is scenario 1. In both the first and third case hospital, the procedures of all doctors are administered in the invoicing system directly. In the second case hospitals, doctors use multiple systems. As a result, changes in for example the risiv codes need to be applied in 1 (case 1 and 3) or n (case 2) applications.

Another observation is the application of design rules. In cases one and two, a design rule for the patient identification code syntax is created, which enables better functional integrations.

In contrast, certain design choices only shift the responsibility for handling a certain change. In scenario 6, it was discussed how the interface between surgical and sterilization departments requires either one or the other department to implement a change in the procedure classification system.

Finally, a remarkable difference was observed in relation to scenario 5 (changes in the patient classification system). In the second and third case hospital, the invoicing process is based on the patient classification, which can be considered as derived data: the classification combines different patient characteristics which result in a similar invoicing category at a certain point in time. However, changes in how certain procedures need to be invoiced will not always be distinguishable in the category classification. This issue has already resulted in manual data tracking. In contrast, the first case hospital bases its invoicing process on raw data. As a result, a direct traceability exists between the invoiced amount and the billable items.

V. Conclusion

The discussion above demonstrates that general engineering insights can be applied directly to a set of relevant domain changes. As such, the generalization of design solutions for domain-specific artefacts which adhere to certain quality characteristic, such as flexibility, should be pursued. While we do not argue that the number of modular dependencies should be considered as a hard quantitative metric, the absence of dependencies, combined with the prioritization of scenarios, enables a rational argument for a certain design to be proposed as a pattern.

While the current research does not yet propose concrete patterns, several contributions to the applied methods can be claimed. A set of open issues in the SAAM method has been identified by [21]. Amongst others, it is argued that SAAM lacks a clear quality metric for architectural attributes, that architecture descriptions are fuzzy notions without a standardized notation, and that SAAM limits itself to the listing of the different steps, omitting to provide techniques to actually perform the steps. Some of these remarks are addressed in this project. For the enterprise architecture field, a clear lack of prescriptive solution has been reported [22]. The elimination of modular couplings in the DSM could lead to such a set of domain-specific principles.

The current state of this research contains various limitations, which align well with the limitations of other scenario-based methods discussed by [8]. First, they argued that the information needed to make fundamental modifiability-related decisions is not necessarily available in documentation. We acknowledge that the determination of, for example, the attributes in the DSM remains largely dependent on the knowledge and experience of the stakeholders. Second, Lassing et al. argue that the actual evolution of a system remains to a large extent unpredictable. As a result, one cannot expect that the list of scenarios is complete, or that every scenario will be implemented. This remains true in our approach. However, the scenarios are first and foremost the means to an end: namely to provide a starting point to discover modular dependencies Third, architectural changes often concern complex components, and this complexity might not be known at the architecture level. In our approach, the granularity of the modules is very coarse. Capturing all complexities and interactions would require a very large DSM. Different techniques might need to be explored to fulfill this role.

References

[1] R. Hausmann, C. A. Hidalgo, S. Bustos, M. Coscia, A. Simoes, and M. A. Yldrm, The Atlas of Economic Complexity: Mapping Paths to Prosperity. MIT Press, 2013.

[2] A. Jalali, "Aspect-oriented business process management," Ph.D. dissertation, Stockholm University, 2016. [Online]. Available: https://su.diva-portal.org/smash/get/diva2:1044437/FULLTEXT01.pdf

	New medical cabinet supplier (S)	**Introduction new medical specialization (S)**	**Opening a new site (S)**	**Changing risiv code (S)**	Education personnel	Amount of experts needed for integration	**Changes in the patient classification system (S)**	**Changes in the procedure classification system (S)**	Tool classification	Flexibility personnel	Amount of registration desks	**Changes in the patient registration process (S)**	Amount of versions patient identification process	Patient identification process	Application choise	Patient identification data	Amount of applications	Integration application and HL7 bus	Amount of applications involved in integration proces	Invoicing module	Integration operation room and sterilization	Operation room system	Appointment system	Sterilization system	Extra data entity 'site'	Permissions
New medical cabinet supplier (S)																										
Introduction new medical specialization (S)																										
Opening extra site (S)																										
Changing risiv code (S)																										
Education personnel							1,2					1,2,3		1,2,3	1,2,3											
Amount of experts needed for integration																			1,2							
Changes in the patient classification system (S)											1,2															
Changes in the procedure classification system (S)																										
Tool classification																										
Flexibility personnel													1,2,3				1,2,3									
Amount of registration desks		1,2																								
Changes in the patient registration process (S)											1,2,3					1,2,3										
Amount of versions patient identification process															1,2,3											
Patient identification process																1,2,3										
Application choise	1,2,3	1,2																								
Patient identification data															1,2,3											
Amount of applications																										
Integration application and HL7 bus															1,2	1,2										
Amount of applications involved in integration process																										
Invoicing module							1,2									1,2,3										
Integration operation room and sterilization								1	3																	
Operation room system								1								1										
Appointment system								3								3										
Sterilization system									3																	
Extra data entity 'site'		3	1,3																							
Permissions		2	2																							

Figure 1. Cross-Case Design Structure Matrix.

[3] D. Campagnolo and A. Camuffo, “The concept of modularity in management studies: A literature review,” International Journal of Management Reviews, vol. 12, no. 3, September 2010, pp. 259–283.

[4] R. Silvestro and P. Lustrato, “Exploring the “mid office” concept as an enabler of mass customization in services,” International Journal of Operations & Production Management, vol. 35, no. 6, 2015, pp. 866–894.

[5] A. Lindstrom, “On the syntax and semantics of architectural principles,” in Proceedings of the 39th Annual Hawaii International Conference on System Sciences (HICSS2006), vol. 8, 2006, pp. 178b–178b.

[6] R. Kazman, L. Bass, M. Webb, and G. Abowd, “Saam: A method for analyzing the properties of software architectures,” in Proceedings of the 16th International Conference on Software Engineering, ser. ICSE ’94. Los Alamitos, CA, USA: IEEE Computer Society Press, 1994, pp. 81–90.

[7] R. Kazman, M. Klein, M. Barbacci, T. Longstaff, H. Lipson, and S. Carriere, “The architecture tradeoff analysis method,” Software Engineering Institute, Carnegie Mellon University, Pittsburgh, PA, Tech. Rep. CMU/SEI-98-TR-008, 1998.

[8] N. Lassing, P. Bengtsson, H. van Vliet, and J. Bosch, “Experiences with alma: Architecture-level modifiability analysis,” J. Syst. Softw., vol. 61, no. 1, Mar. 2002, pp. 47–57.

[9] M. R. Davoudi and F. S. Aliee, “Characterization of enterprise architecture quality attributes.” in EDOCW. IEEE Computer Society, 2009, pp. 131–137.

[10] L. Bass, P. Clements, and R. Kazman, Software Architecture in Practice, 3rd ed. Addison-Wesley Professional, 1998.

[11] P. Clements, R. Kazman, and M. Klein, Evaluating Software Architectures: Methods and Case Studies. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 2002.

[12] P. Johnson, E. Johansson, T. Sommestad, and J. Ullberg, “A tool for enterprise architecture analysis,” in 11th IEEE International Enterprise Distributed Object Computing Conference (EDOC 2007), Oct 2007, pp. 142–142.

[13] J. W. Ross, P. Weill, and D. C. Robertson., Enterprise Architecture as Strategy – Creating a Foundation for Business Execution. Harvard Business School Press, Boston, MA, 2006.

[14] J. A. Zachman, “A framework for information systems architecture,” IBM Systems Journal, vol. 26, no. 3, 1987, pp. 276–292.

[15] The Open Group, “The open group architecture framework (togaf) version 9,” 2009. [Online]. Available: http://www.opengroup.org/togaf/ [Accessed: 31-01-2017]

[16] O. Noran, Handbook on Enterprise Architecture. Springer-Verlag, 2003, ch. A mapping of individual architecture frameworks (GRAI, PERA, C4ISR, CIMOSA, Zachman. ARIS) onto GERAM, pp. 65–210.

[17] M. Schenherr, “Towards a common terminology in the discipline of enterprise architecture.” in ICSOC Workshops’08, 2008, pp. 400–413.

[18] A. Cabigiosu and A. Camuffo, “Beyond the “mirroring” hypothesis: Product modularity and interorganizational relations in the air conditioning industry,” Organization Science, vol. 23, no. 3, May/June 2012, pp. 686–703.

[19] O. Becker, J. Ben-Asher, and I. Ackerman, “A method for system interface reduction using n^2 charts,” Systems Engineering, vol. 3, 2000, pp. 27–37.

[20] C. Y. Baldwin and K. B. Clark, Design Rules, Volume 1: The Power of Modularity, ser. MIT Press Books. The MIT Press, January 2000.

[21] M. T. Ionita, D. K. Hammer, and H. Obbink, “Scenario-based software architecture evaluation methods: An overview,” in Workshop on methods and techniques for software architecture review and assessment at the international conference on software engineering, 2002.

[22] T. Tamm, P. B. Seddon, G. Shanks, and P. Reynolds, “How does enterprise architecture add value to organisations?” Communications of the Association for Information Systems, vol. 28, no. 1, 2011, pp. 141–168.