Architecture Principles Compliance Analysis

João Pedro Marques Alves

Thesis to obtain the Master of Science Degree in

Information Systems and Computer Engineering

Supervisors: Prof. André Ferreira Ferrão Couto e Vasconcelos
Prof. Pedro Manuel Moreira Vaz Antunes de Sousa

Examination Committee

Chairperson: Prof. José Manuel Nunes Salvador Tribolet
Supervisor: Prof. André Ferreira Ferrão Couto e Vasconcelos
Member of the Committee: Prof. Artur Miguel Pereira Alves Caetano

June 2014

Acknowledgments

A realização deste trabalho não poderia ter sido alcançada sem a contribuição de várias pessoas, que de uma forma ou de outra, foram essenciais para a sua conclusão.

Antes de tudo, quero agradecer ao meu orientador, o Professor André Vasconcelos, por todo o apoio, sugestões, paciência e motivação para completar o trabalho. Quero agradecer igualmente ao meu co-orientador, o Professor Pedro Sousa, pela disponibilidade demonstrada ao longo da dissertação.

À Fidelidade, por me fornecer a oportunidade de desenvolver este trabalho em contexto empresarial, o que se apresentou como uma experiência enriquecedora a nível pessoal e profissional. Quero, igualmente agradecer, à equipa de Arquitectura da Fidelidade por toda a disponibilidade, ajuda e simpatia demonstrada, em especial à Cristina Arbués.

A todos os meus amigos, quero pedir desculpa pela minha ausência em alguns momentos importantes, mas também tenho de agradecer toda a amizade mesmo assim mostrada. Neste caso, tenho de endereçar um agradecimento especial aos meus companheiros, Fábio Ferreira e Hugo Ramos por todas as directas, discussões e paródias vividas ao longo do curso que representam uma boa amizade.

À minha família, Mãe, Pai, Irmã, Avó e à Andreia, todas as palavras são poucas para poder descrever, o quão agradecido estou por tudo o que fizeram e fazem por mim. Pelo que significam para mim, apenas espero retribuir o que me dão, e dar-vos motivos para terem orgulho em mim. Aos meus avós, que já não estão entre nós fisicamente, apenas quero enaltecer toda a gratidão e amor que tenho por cada um e que de certo, tiveram também uma boa quota parte na motivação para realizar este trabalho.

A todas as pessoas, endereço novamente um agradecimento especial, pois sem elas, este trabalho certamente não seria alcançado.

Abstract

The architecture principles play a key role in the enterprise architecture evolution. However, the architecture does not always address the principles intentions, which could result in unplanned deviations. So, it's evident that organizations need to possess a mechanism to analyze whether their architecture design is or not compliant with their architecture principles.

Through the related work study, it's perceptible the lack of an architecture analysis that enables to evaluate the architecture compliance regarding the architecture principles. To surpass the referred lack, this research proposes an architecture analysis which allows evaluating the architecture compliance with its guiding principles.

An architecture principle could be considered the rationale for the presence of several elements and relationships in the architecture, which enables to characterize a principle through its expected impact. The proposed analysis consists in the principle expected impact recognition to evaluate the architecture, which consequently enables to identify its compliant elements. This impact is formalized, based on ArchiMate, which enables to analyze the architecture through its enterprise architecture descriptions. The proposed analysis includes a dedicated analysis for nine principles. These principles are sufficiently objective to be characterized from their architecture impact.

To demonstrate the research proposal, it is applied to analyze the compliance of some specific architectures. These architectures, provided by a Portuguese insurance company, should address the considered principles instructions. Thus, regarding the guiding principles for each architecture are identified its compliant elements through the proposed analysis. Hereupon, the proposal feasibility positions this research as a contribution to the architecture principles field.

Keywords: Enterprise Architecture, Architecture Principles, Architecture Analysis, Architecture Principle Compliance Analysis.

Resumo

Os princípios arquitecturais desempenham um papél crucial na evolução da arquitectura empresarial. Contudo, a arquitectura nem sempre endereça as intenções dos princípios, o que pode resultar em desvios inesperados. Assim, é evidente a necessidade das organizações em possuir um mecanismo para analisar se o seu desenho arquitectural está conforme com os princípios arquitecturais.

Através do trabalho relacionado, é perceptível a falta de uma análise arquitectural que permita avaliar a conformidade da arquitectura em relação aos princípios. Para colmatar a referida lacuna, esta investigação propõem uma análise arquitectural que permite avaliar a conformidade da arquitectura com os seus princípios orientadores.

Um princípio arquitectural, pode ser considerado o racional para vários elementos e relações na arquitectura, o que permite caracterizar um princípio através do seu impacto esperado. A análise proposta, consiste no reconhecimento do impacto do princípio para avaliar a arquitectura, o que consequentemente permite identificar os elementos conformes. Este impacto é formalizado, baseado em ArchiMate, o que permite analisar a arquitectura através das suas descrições. A análise proposta inclui uma análise dedicada para nove princípios, sendo estes suficientemente objectivos para serem caracterizados pelo seu impacto arquitectural.

Para demonstrar a proposta de investigação, esta é aplicada para analisar a conformidade de algumas arquitecturas específicas. Estas arquitecturas, fornecidas por uma seguradora Portuguesa, devem obedecer às instruções dos princípios considerados. Assim, para os princípios relativos a cada arquitectura, são identificados os elementos conformes através da análise proposta. Assim, a viabilidade da proposta, posiciona esta investigação como uma contribuição para o campo dos princípios arquitecturais.

Keywords: Arquitectura Empresarial, Princípios Arquitecturais, Análise Arquitectural, Análise da Conformidade de Princípios Arquitecturais.

Contents

List of Tables

List of Figures

List of Acronyms

EA – Enterprise Architecture

EAA – Enterprise Architecture Analysis

EAAs - Enterprise Architecture Analyses

EAD – Enterprise Architecture Description

EADs – Enterprise Architecture Descriptions

PPR – Retirement Savings Plan

EAM – Enterprise Architecture Management

FOSBO - Front-Office Processes Are Separated from Back-Office Processes

CSSCI - Channel-Specific Is Separated from Channel-Independent

DPS - Data Are Provided by the Source

DMSA - Data Are Maintained in The Source Application

AAM - Applications Are Modular

AFTEP - Application Functionality is Available Through an Enterprise Portal

PPBLS - Presentation Logic, Process Logic and Business Logic Are Separated

ITSCTS - IT Systems Communicate Through Services

IESLD - Integration with External IT Systems Is Localized in Dedicated IT Components

Chapter 1

Introduction

Modern day enterprises are faced with a range of challenges imposed by their environment (Op't Land et al., 2008). These challenges have impact in the way that organizations hold their evolution and reach their strategy, making them transform.

This is where enterprise architecture (EA) is positioned by the organizations as an instrument to coordinate and steer their transformation (Greefhorst and Proper, 2011). The EA guides enterprise future direction providing a better base for decision making (Op't Land et al. 2008), while enables the achievement of organizational cohesion and integration (Lankhorst, 2009; The Open Group, 2009). The EA design goes from the definition of products and services offered to the clients, via the business processes that deliver these products and services, and the information systems that support these processes, to the underlying IT infrastructure (Greefhorst and Proper, 2011). The robustness of this design is critical to face the imposed challenges.

Hereupon, the EA design must evolve in order to make effective the organization adaption to the environment. To properly guide this evolution, the architecture principles are positioned as the key ingredient. The architecture principles provide rules and guidelines to inform and support the way in which an organization sets about fulfilling its mission (The Open Group, 2009). Therefore, it's important that EA design complies with their guiding principles, which is not always achieved. This emphasizes the need for an EA compliance evaluation based on architecture principles.

However, the related work study shows that an EA analysis to evaluate the EA compliance with their guiding principles still lack. Hereupon, our vision pretends to formalize architecture principles, based on ArchiMate to enable their EA compliance analysis. This formalization enables to analyze an enterprise architecture description (EAD) through the detection of architecture structures that represent the principle expected impact, and consequently identify their compliant elements.

1.1 Research Problem

Associated with the outlined strategy, there is a target architecture that should be reached. This is where the architecture principles give advice on how to design the pretended architecture by restricting the design freedom of EA projects (Hoogervorst, 2004; Hoogervorst, 2009). The architecture principles could also be considered the rationale for the presence of certain elements and relationships in the EA (Greefhorst and Proper, 2011).

As stated by (Greefhorst and Proper, 2011), the principles effectiveness is closely linked how the respective stakeholders assimilate and interpret them. So, as a result of the principles application is expected the stakeholders' decision-making aligned with the principles intentions, to be possible the achievement of the desired architecture (Lindström, 2006).

However, there are several factors that could endanger the proper principles interpretation. The ambiguity that could exist in the principle specification, the organizational context where the principle is being applied and the lack of precise information in the principle specification, represent some of the existent factors that could influence the stakeholders' interpretation (Greefhorst and Proper, 2011). Thus, a misinterpretation may imply deviations from the principle intended impact in the EA, which could result in the organizations failure to achieve their strategic objectives (Greefhorst and Proper, 2011). Hereupon, it's evident that organizations need to possess a mechanism to analyze whether their architecture design is or not compliant with their guiding architecture principles.

In the architecture analysis field, the EA models analysis plays a central role (Lankhorst, 2009). The enterprise holistic view provided by EADs, which could be adapted to the stakeholders concerns, present the EADs as artifacts with valuable information to plan and analyze the EA evolution. The EADs can endorse existing, transition or target architectures which could be useful to evaluate if the architecture evolution is being properly conducted (Šaša and Krisper, 2011). However, through the literature study, none of the existing architecture analysis, based on EADs, addresses the architecture compliance with the architecture principles.

Thus, regarding the relevance that EADs can play and the organizations need to assess the principle compliance, becomes evident that the non-existence of an architecture analysis that combines these two points is a substantial gap in the literature. This non-existent architecture analysis corresponds to the problem endorsed by this research.

Hereupon, when we try to perceive how and what should be considered to approach the identified gap, several questions emerge. The following questions address the research questions to be answered by this work.

Q1: What are the existing architecture analyses that could be used to enable the architecture principle compliance analysis through EADs?

Q2: The EADs could be used to verify the architecture compliance with its guiding principles?

Through this last question other related questions arise.

Q3: What architecture elements and relationships are impacted by the architecture principles and should be analyzed in the compliance analysis?

Q4: What are the conditions to be followed by the impacted architecture elements and relationships to be compliant with the respective principle?

To summarize, the problem identified is the **lack of an architecture analysis to evaluate the architecture compliance with their guiding principles, through the respective EADs**. To address this problem, this research will seek to answer to the questions defined above.

1.2 Contributions

This research aims to evolve the architecture principle field through the evolution of the architecture evaluation field. To surpass the identified problem and answer to the respective research questions, this work will present the following contributions:

- (Main) The proposal of an architecture analysis to evaluate the EA compliance with its guiding principles. This analysis consists in the recognition of architecture structures that represent the principle expected impact. This recognition is based on the formalization of these structures in ArchiMate, which enables to identify the compliant elements in the respective EADs.
- (Secondary) The proposal of a mechanism to perform the compliance analysis previously mentioned in an automatic way. This mechanism consists in the enterprise architecture management (EAM) tool parameterization to recognize the principle expected impact, and consequently recognize the principle compliant elements. This mechanism enables to apply the principle compliance analysis to a broader and complex scope.
- (Secondary) To demonstrate the compliance architecture analysis, it's used a case study provided by Fidelidade. This case study besides being used to demonstrate the proposal feasibility is also valuable to the architecture principles field due to its novelty.
- (Secondary) To understand what could be addressed by this investigation, the literature study related with architecture principles and architecture analysis are presented. This study presents itself as a contribution due to non-existence of literature that relates these two fields.

1.3 Research Methodology

The Design Science Research Methodology (DSRM) has been considered to properly guide this research. The design-science paradigm is fundamentally a problem solving paradigm. Regarding the Information System research, the design science creates and evaluates artifacts intended to solve identified organizational problems. The artifacts that are built to address these problems are then evaluated with respect to the utility provided in solving those problems. So, the artifacts produced by the DSRM could be classified as constructs (vocabulary and symbols), models (representations), methods (algorithms and best practices) and instantiations (implementations and prototypes) (Hevner et al., 2004).

Based on the artifacts produced by DSRM, the artifact proposed by this research could be characterized as a model. Models aid problem and solution understanding and frequently represent the connection between problem and solution, enabling exploration of the effects of design decisions and changes in the real world (Hevner et al., 2004).

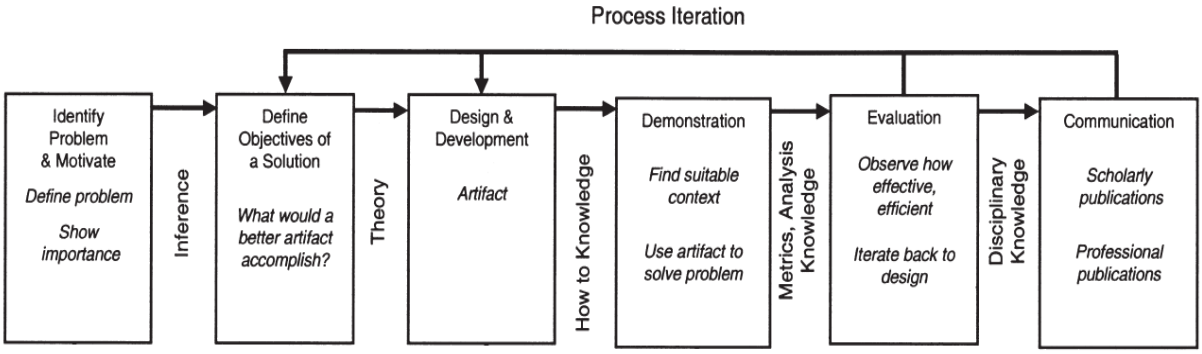

Figure 1: DSRM process, adopted from (Peffers et al., 2008).

The DSRM consideration implies the incorporation of required principles, practices and procedures to carry out such research. So, Peffers et al. (2008) propose an iterative process (Figure 1) that enables the appliance of this methodology through the following sequence of activities:

1) **Problem identification and motivation.** Define the specific research problem and justify the value of the solution. In this activity the relevance of the research problem is highlighted to motivate the researcher and the audience to pursue the solution.
2) **Define the objectives for a solution.** Infer the objectives of a solution from the problem definition and knowledge of what is possible and feasible.
3) **Design and development.** This activity includes determining the artifact's desired functionality and its architecture and then creating the actual artifact.
4) **Demonstration.** Demonstrate the use of the artifact to solve one or more instances of the problem. This could involve its use in experimentation, simulation, case study, proof, or other appropriate activity.
5) **Evaluation.** Observe and measure how well the artifact supports a solution to the problem. This activity involves comparing the objectives of a solution to actual observed results from use of the artifact in the demonstration. It requires knowledge of relevant metrics and analysis techniques. At the end of this activity the researchers can decide whether to iterate back to the "Design and development" activity.
6) **Communication.** Communicate the problem and its importance, the artifact, its utility and novelty, the rigor of its design, and its effectiveness to researchers and other relevant audiences such as practicing professionals.

This research will endorse all the activities sequence proposed by this process, which will be followed by the document structure.

1.4 Document Structure

As previously mentioned this research was conducted based on the DSRM, which is evident by the structure adopted in this document. The document structure is composed as follows:

- **Chapter 1.** The motivation, the research problem, the related contributions and the research methodology underlying this research are presented.
- **Chapter 2.** In this chapter, the related work is highlighted. It is here that are provided the foundations to perceive the domains underlying the identified problem and the proposed solution.
- **Chapter 3.** In this chapter the solution proposal is scrutinized and it's where is explained the approach underlying the proposal. This chapter also endorses the solution proposal adapted for each architecture principle.
- **Chapter 4.** In this chapter is presented a case study provided by Fidelidade, where are applied and demonstrated the proposed solution feasibility.
- **Chapter 5.** The obtained results from the case study are evaluated, where it's verified if the research problem and contributions proposed by this work are or not properly achieved.
- **Chapter 6.** In this chapter, it is summarized what was considered during this research. There are highlighted the resulting contributions and limitations from this work. There are also presented future directions that could be followed based on this research.

Below, the Table 1 shows a detailed correlation between the DSRM steps and the considered document structure.

DSRM Activity	Chapter
Problem identification and motivation	1.1 - Research Problem 2 – Related Work
Define the objectives for a solution	1.2 - Contributions
Design and development	3 – Proposal
Demonstration	4- Case Study
Evaluation	5 - Evaluation
Communication	6.1 – Communication

Table 1: Relation between the DSRM and document structure.

Chapter 2

Related Work

In this chapter, are analyzed in the literature the domains related with this research. This analysis will supports the research problem identification, but also enables to understand what the existing tools to solve it are. Initially, is introduced the EA theme and then are presented two main themes. One of them presents the state of architecture principle field and the other the existing architecture analyses to evaluate the EA quality.

2.1 Enterprise Architecture

Is stated by (Lankhorst, 2009) that's difficult to an organization achieve business success without a good architecture. The EA can be described as a coherent whole of principles, methods, and models that are used in the design and realization of an enterprise's organizational structure, business processes, information systems, and infrastructures (Lankhorst, 2009).

The EA main purpose is to align the design of an enterprise to its strategy providing a normative restriction of design freedom toward transformation projects and programs (Greefhorst and Proper, 2011). It enables a better decision making by sharing knowledge on architecture decisions and provide also a way to describe and control an organization's structure, processes, applications, systems, and technology in an integrated way (Lankhorst, 2009).

An EAD is usually very complex, because it comprises a large set of components and relationships between them. Although, the EA can be represented with the relevant information for the stakeholder concerns and for that reason viewpoints and views are defined related with different stakeholders (Šaša and Krisper, 2011). A viewpoint in ArchiMate is defined as a selection of a relevant subset of elements and relationships that the corresponding views should comprise (The Open Group, 2012).

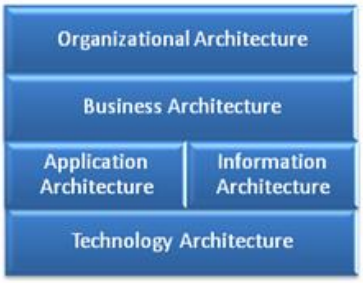

Figure 2: EA composition. (Gama et al., 2007)

Based on (Gama et al., 2007), the EA composition is characterized by the following architectures (Figure 2):

- The **organizational architecture** addresses organizational aspects not directly related to the specifics of the business or with the mechanisms used in the creation of value (Vasconcelos,

2007). It involves aspects such as the hierarchical structure, organizational culture, tacit and implicit relationships that govern the internal relations of the organization, the roles of different actors, rules, objectives and fundamentally human resources (Gama et al., 2007).

- The **business architecture** describes how the organization functionally operates. It is responsible for defining the business processes and goals that are needed to implement the organization strategy (Vieira et al., 2004).
- The **application architecture** automates business processes needs by making use of informational entities necessary for its operation, thereby resulting on the relationship business processes/information (Gama et al., 2007). It provides a blueprint for the individual applications to be deployed, their interactions, and their relationships to the core business processes of the organization (The Open Group, 2009).
- The **informational architecture** aims to identify and define the main types of data that support the business development of the organization. It's through this architecture that are described the informational entities necessary to the development of the business processes (Vasconcelos, 2007).
- The **technology architecture** describes the logical software and hardware capabilities that are required to support the deployment of business, data, and application services (The Open Group, 2009). It basically consists in selecting the technologies to be used as support of the systems and applications defined in the application and information architectures (Gama et al., 2007).

2.2 Enterprise Architecture Principles

The architecture principles can be seen as general rules and guidelines, intended to be enduring and seldom amended, that inform and support the way in which an organization sets about fulfilling its mission (The Open Group, 2009). They play a prominent role in the development and advancement of the EA, giving advice how to design target architecture by restricting the design freedom of EA transformation projects (Aier et al., 2011). They also provide motivations for fundamental design decisions and fundamental design knowledge (Greefhorst and Proper, 2011).

The architecture principles as a means to direct the EA require their specification (Greefhorst and Proper, 2011; Richardson et al., 1990). In Table 2 is highlighted the structure that is generally accepted by the literature to specify a principle.

Statement	Should succinctly communicate the fundamental rule.
Rationale	Should highlight the business benefits of adhering to the principle.
Implications	Should highlight the requirements for carrying out the principle.

Table 2: Basic structure to specify an architecture principle. (Greefhorst and Proper, 2011)

Beside the attributes proposed above, other attributes could be used in the specification process (Greefhorst and Proper, 2011; Van Bommel et al. 2006; Chorus et al., 2007; Aitken, 2010). These

attributes could be used to increase the principle semantics and to make easier the compliance to them.

The architecture principles to be really effective and consequently be considered good principles they must have a clear semantic, an understandable syntax and the right focus (Lindström, 2006; Van Bommel et al., 2007). However, if any of these characteristics are violated some deviations in the expected impact could emerge (Greefhorst and Proper, 2011). This fact emphasizes the need to verify if the impact on the EA is the impact prescribed by the architecture principles. To understand how this verification could be done, two aspects have to be analyzed. The first, is how the architecture principles are applied and then, is how are managed their compliance. Both aspects are considered in the process proposed by Greefhorst and Proper (2011) to handle the principle lifecycle.

Relatively to the principle application is specially emphasized the transformation activity, which is separated in two types. The first one is called derivation and consists in the principles transformation into statements that are relevant in a more specific context. For example, architecture principles from upstream architectures are transformed into requirements for specific solution architecture. The other transformation is related with the architecture principles transformation to models. This transformation is applied to models at various levels that go from EA models, solution architecture models to design models. This transformation it´s built on the fact that architecture principles could be the rationale behind a number of elements in the model and for their relationships (Greefhorst and Proper, 2011).

Hereupon, it's important to relate the transformations with the respective compliance management. In the compliance management is advised the architecture principle refinement into requirements and then in design decision to proceed to the compliance verification. However, this advice maps with the derivation transformation. Regarding the compliance management, is evident a lack of a compliance verification to the transformation of the principles to models and it is here, that our work presents itself as a contribution.

Finally, it's important to perceive what architecture principles could be used in this work. In the literature two catalogues with architecture principles are presented by Greefhorst and Proper (2011) and TOGAF (The Open Group, 2009). The principles specified in both catalogues obey the structure previously defined, however in (Greefhorst and Proper, 2011) are used also other attributes. The additional attributes identify the architecture domains that are impacted by the principle and what are the quality characteristics expected by its application. Below in Table 3, is presented an architecture principle selected from the catalogue presented in (Greefhorst and Proper, 2011).

A.6 Management Layers Are Minimized
Type of information: business **Quality attributes:** reliability, usability, efficiency, maintainability
Rationale: • Elimination of management layers minimizes overhead costs. • By eliminating management people tend to take more responsibility for their work, which

increases the quality and efficiency.
Implications: • There are as few layers of management as possible. • The ultimate objective is to have self-directed teams throughout an organizational unit with no layers of management at all. • People who perform the actual work have responsibility for making decisions.

Table 3: Architecture principle example extracted from (Greefhorst and Proper, 2011).

Critical Analysis

Due to the architecture principles novelty, techniques that enable their compliance analysis through an EA model still lack. This fact evidences the non-consideration of EADs in the principle compliance management, and it's here where this work pretends to present itself as a contribution. This novelty is also perceptible with the small number of architecture principles in the literature (Vieira, 2012).

As mentioned before, the architecture principle specification could not be sufficient specific or objective, which could result in the stakeholders' non-perception of what is the principle impact. The principle presented in Table 3, represents one example where it's difficult to deduce what the real impact of it in the EA is. This difficulty is due to the lack of objectivity in the principle specification and for that reason could be difficult to evaluate its compliance. So, not all architecture principles could be used to be evaluated through an EAD, which requires a careful choose of the architecture principles considered in this research.

As previously referred the existing principles in the literature are endorsed by two different catalogues, one of them is provided by Greefhorst and Proper (2011) and the other by The Open Group (2009). In this case a comparison between the two catalogues is needed to verify from what catalogue the principles should be considered. So, the principles available by Greefhorst and Proper (2011) present some advantages:

- They are based on real-world architectures (Greefhorst and Proper, 2011);
- More level specific (Vieira, 2012);
- They are more complete principles. They are not restricted to an architecture domain but also present the propagations that could exist in the other architecture domains.

Hereupon, the principles endorsed by this research will be selected from the Greefhorst and Proper (2011) catalogue.

2.3 Enterprise Architecture Modeling

The analysis intended by this research is intimately connected how EA could be represented. So, in this section the ArchiMate modeling language is introduced and the reasons for its consideration are also highlighted.

2.3.1 ArchiMate

The ArchiMate is an EA modeling language that provides a uniform representation for diagrams that describe EA. The ArchiMate offers an integrated architectural approach that describes and visualizes the different architecture domains and their underlying relations and dependencies (Lankhorst, 2009).

The main motivation for development of this language was to surpass the gap between different architecture domains, for example business process domain, information systems domain and technical domain (Lankhorst, 2004). Before the ArchiMate creation, the architects had to rely on existing methods and techniques from disparate domains, for example, the BPMN notation (OMG, 2009) for the business process domain, and UML notation (Staron et al., 2006) for the information systems domain (Lankhorst, 2009). The different architecture domains identified by Lankhorst (2009) are organization, product, process, application, information and technical domains.

The core language consists on the three following types of elements (The Open Group, 2012):

- The active structure elements are defined as entities that are capable of performing behavior.
- The behavior elements are defined as units of activity performed by one or more active structure elements.
- The passive structure elements are defined as the objects on which behavior is performed.

In Figure 3 is presented the generic ArchiMate metamodel with the core concepts used to describe each architecture domain. The active structure elements are represented in green, the behavior elements in yellow and the passive structure elements in blue.

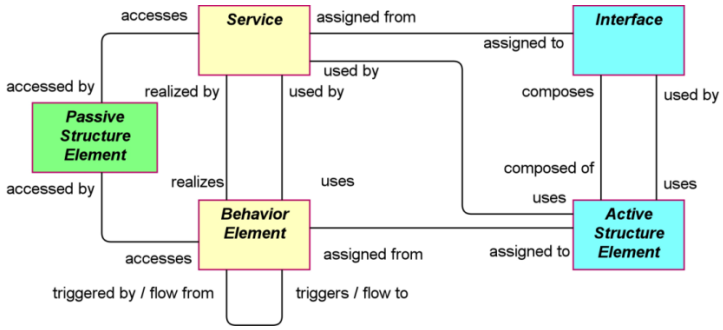

Figure 3: The core concepts of ArchiMate. (The Open Group, 2012)

The completed ArchiMate metamodel with all elements and relationships that could be used to represent an EA is presented in Figure 4. This metamodel contemplates all elements and relationships used to represent all architecture domains and the relationships between them.

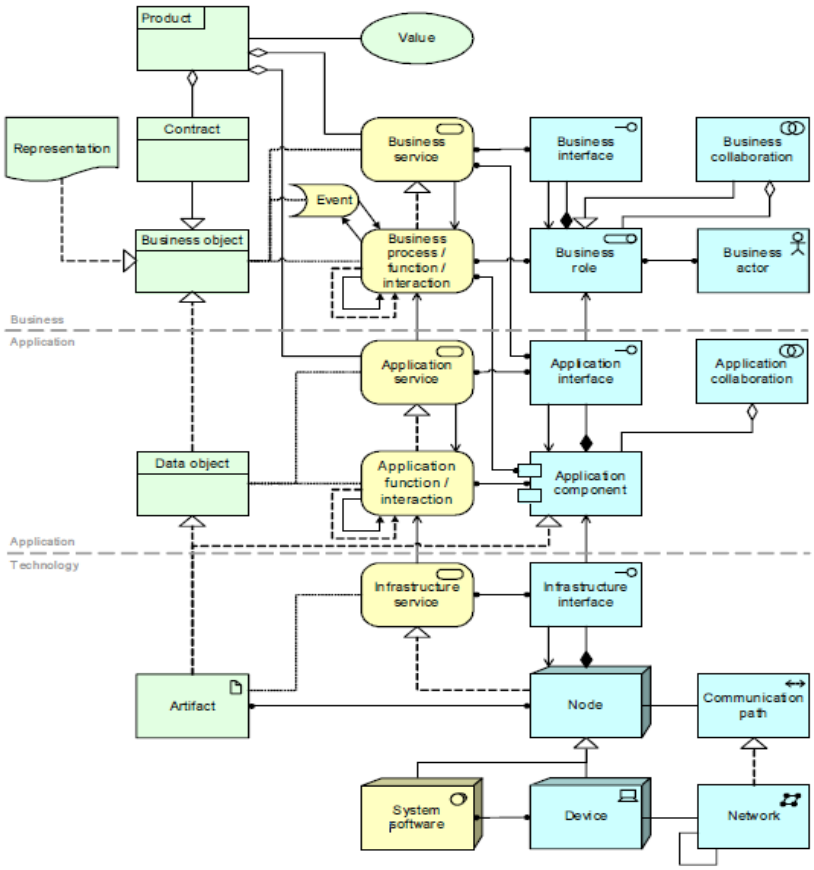

Figure 4: Overview of the ArchiMate concepts and main relationships. (Lankhorst, 2009)

How it's noticeable in the previous figure, the ArchiMate defines three main layers (The Open Group, 2012):

- The Business Layer offers products and services to external customers, which are realized in the organization by business processes performed by business actors.
- The Application Layer supports the business layer with application services which are realized by (software) applications.
- The Technology Layer offers infrastructure services (e.g., processing, storage, and communication services) needed to run applications, realized by computer and communication hardware and system software.

Therefore, the ArchiMate core language, embedded in the ArchiMate metamodel contains only the basic concepts and relationships that serve general enterprise architecture modeling purposes. Although, the language through extensions mechanisms enables to perform specific types of model analysis, the architecture communication or even to specialize the language itself. These extension mechanisms could be the addition of attributes to ArchiMate concepts and relationships or through the specialization of concepts (The Open Group, 2012).

Critical Analysis

Besides being a modeling language recognized as standard by the Open Group there are other reasons behind the ArchiMate consideration in this research. These reasons are the follows.

- The ArchiMate metamodel is aligned with the architecture principles selected from the catalogue referred in Section 2.1 (Vieira, 2012). It's possible to match the components and relationships referred by the principles with the ArchiMate concepts.
- The homogeneity between different architectures enables to avoid the consideration of specific languages for each domain. This homogeneity turns easier to evaluate the principle impact that propagates through multiple domains. The same happens when the principle impact endorses relationships between domains.
- Enables the creation of EADs as viewpoints and views. This enables to provide the insight to the respective stakeholders of the architecture elements that correspond to the principle impact.
- It's possible to extend the ArchiMate metamodel (The Open Group, 2012). This issue is important because could be necessary unambiguously identify a certain architecture element or relationship. This need emerge when these elements or relationships don't exist in the metamodel or represent a specification of them.
- Several EA techniques to analyze the EA quality are based on ArchiMate which could represent a good background for this research. These analyses are presented in the next section.

The reasons considered above present themselves as the justification for the use of the ArchiMate language in the proposed solution.

2.4 Enterprise Architecture Analysis

The enterprise architecture analysis (EAA) presents itself as a mechanism to ensure the rationale of the decision-making in the architecture evolution (Šaša and Krisper, 2011; Lankhorst, 2009). Through the literature study the EAA comprises two categories of analysis. One of these categories could be seen at the quantitative level where is evaluated if a certain solution is better than other. The other category could be placed at the qualitative level which enables to detect incoherencies and inconsistencies that possibly could exist in the architecture (Šaša and Krisper, 2011). The EA discipline advocates the use of models to support decision-making (Johnson et al., 2007). The use of models is considered in both categories previously referred.

Lankhorst (2009) describes different architecture analysis techniques that help architects and stakeholders to compare alternative designs and, hence, take well-informed design decisions when making trade-offs between aspects like cost, quality, and performance and to be able to study the impact of a change to the design. In (Lankhorst, 2009) quantitative and functional analysis techniques are distinguished.

Quantitative architecture analysis focuses on the quantitative aspect of relationships between different EA elements and layers. It can be used for optimization by quantifying the effect of alternative design choices, to obtain measures to support impact-of-change analysis, and for capacity planning (Šaša and Krisper, 2011). This analysis on the architecture principle field is related with the usage of metrics to measure the architecture compliance. Several metrics based on architecture principles are presented by Vieira (2012).

Functional or qualitative analysis enables to understand how a system that conforms to the architecture works, to find the impact of a change on the architecture, or to validate the architecture correctness. Relatively to this analysis are distinguished the structural (static) and dynamic (behavior) aspects.

Before explaining in which consist the static and dynamic analysis, three concepts have to be considered, they are the signature, symbolic model and semantic model. The signature could be seen as the concepts by means of which the architecture could be described. The symbolic model consists of its signature that specifies symbolically its structural elements and their relationships. A semantic model is defined as a formal interpretation of the symbolic model.

To perform the static analysis of an architecture, its signature forms the basis. This analysis focuses on the symbolic representation of the structural elements and their relationships, abstracting from other architectural aspects like rationale, pragmatics, and visualization. So, this type of analysis is devoid of any semantic. This kind of analysis is used to determine the impact of a change in the architecture, which implies traverse the architecture and taking into account each relation and its meaning to determine whether the proposed change might propagate. Hereupon, this analysis is based on description logic formalisms. Description logics are knowledge representation languages tailored to express knowledge about concepts and concept hierarchies and for that reason, they correspond to useful formalism to perform this analysis.

For dynamic analysis, techniques based on formal interpretations are used. Dynamic analysis is based on semantic models (Lankhorst, 2009) and enables to detect logical errors. This analysis leads to a better consistency and focuses on logical aspects of the models (Šaša and Krisper, 2011).

Other approaches based on architecture patterns exist in the literature. In TOGAF (The Open Group, 2009), patterns are considered a way of putting building blocks into context, used to describe a re-usable solution to a problem. Building blocks could be seen as the architecture elements and relationships which the respective patterns tell how to use them, when, why and what trade-offs you have to consider.

In (Šaša and Krisper, 2011) are presented EA patterns for business support analysis. A pattern formalization based on ArchiMate represents the basis for the approach underlying this analysis. This formalization enables the detection of architecture structures (architecture elements and relationships) that characterize each pattern. The detection of the referred structures enables to be aware of what could be changed in the EA which could result in the consideration of a new established pattern.

The approach considered is divided in three main steps. The first step consists in the perspective (viewpoint) definition, where the elements and relationships that have to be considered in the respective pattern are highlighted. The second step defines the characteristics to be addressed by the architecture elements presented in the views that describe the architecture. These views are based on the perspectives previous defined. These characteristics are then used to recognize the architecture structures that correspond to each pattern. The last step corresponds to the establishment of a mechanism to automatically recognize the referred characteristics. The recognized patterns are then used in the comparison with other formalized patterns to identify how the architecture should evolve relatively to the business support.

Critical Analysis

The solution proposed by this research, pretends to evaluate the EA through the identification of several architecture structures (elements and relationships), if they are compliant with their guiding principles. Hereupon, this research could be positioned in the qualitative analysis. More specifically, it could be considered in the structure analysis, however it is not intended to analyze the change impact in the EA. The proposed analysis enables to evaluate structural aspects of the architecture that are prescribed by the architecture principles. It allows evaluating the architecture coherence, which could result in the improvement of the architecture dynamic evaluation. These improvements could represent the rationale underlying the prescribed architecture principle.

The architecture principles and patterns share the point of being the rationale for the presence of several architecture elements and relationships in the EA. This is the main reason to consider the approach used by (Šaša and Krisper, 2011) in this research. So, the steps defined by the referred approach are related with the architecture principles compliance analysis as follows:

- The architecture principles have a clear impact in some elements and relationships. This fact match with the first step of the (Šaša and Krisper, 2011) approach, where are defined the viewpoints with the elements and relationships to be considered in the respective analysis.
- The architecture principles prescribe several conditions that have to be addressed by the identified architecture elements. The same happens with the patterns recognition and for that reason the second step also maps out.
- The third step also could be considered in our proposal due to the complexity of certain EADs and for that reason an automatic analysis turn easier to analyze a large and complex scope.

Hereupon, the EAAs considered by the previous section, helps to positioning the proposed analysis and understand of what could be behind of its development. It's also important to notice that the EAAs presented also consider the ArchiMate to represent and evaluate the EA.

Chapter 3

Proposal

To surpass the research problem previously identified, this work proposes an architecture analysis based on architecture principles. This analysis enables to identify the principle compliant elements, in the respective EADs, with the EA guiding principles.

The architecture principles are considered the rationale for the existence of several EA elements and relationships (Greefhorst and Proper, 2011), which enable to characterize them through their expected impact in the EA. This characterization represents the basis underlying the proposed analysis.

In this analysis, the architecture structures concerning each principle are recognized when a certain architecture is analyzed. If these structures are recognized, the compliant elements are identified. In the opposite side, if the recognition is not effective no elements are identified as compliant.

So, the analysis proposal endorses the identification of the elements and relationships impacted by each principle. It also addresses how the impacted elements should relate with each other in order to not violate the principle intentions. This last consideration represents the construction of the architecture structures that represent the principle impact.

Hereupon, it's important to know how the architecture principles and the analyzed architecture could be described. In this analysis, the ArchiMate modelling language will be used for that purpose.

3.1 Approach

In this section is presented the approach behind the proposed analysis. This approach, based on (Šaša and Krisper, 2011), initially consists in the identification of the EA elements and relationships needed to perform the principle compliance analysis. Then, the architecture structures that represent the principle expected impact are recognized. This recognition enables to determine the compliant elements in the analyzed EAD.

Hereupon, the used approach is applied for each principle and includes the following steps.

- **To define relevant EA perspectives.** These perspectives can be seen as the necessary viewpoints that represent the elements and relationships impacted by the principle. The goal of such viewpoints, is to ensure that the corresponding views illustrate exactly the relevant elements for the compliance analysis.
- **To define characteristics that address the principle perspectives.** These characteristics define the conditions, prescriptions imposed by the principle. They enable to recognize the principle expected impact in the EAD and consequently identify the compliant elements.

- **To establish a mechanism to automatically recognize the characteristics prescribed by the principle.** This automatic recognition, through an EA management tool, enables to proceed to a larger and complex analysis.

For each principle analysis the two first steps are addressed in the research proposal (Section 3) and the third in the Case Study (Section 4).

In summary, we represent an architecture principle as a set of elements, which is formalized with its membership conditions. If an EA element respects the principle membership conditions it is compliant. The principles definition in their membership set conditions is formalized in a way which enables their implementation using an EAM tool.

Regarding the third step of the approach previously presented, it's relevant to understand what requirements an EAM tool should address to perform the proposed compliance analysis. The consideration of a certain EAM tool to perform the proposed compliance analysis requires the following requirements:

- It has to be compatible with the ArchiMate metamodel;
- It should allow the ArchiMate metamodel extention;
- It must provide a knowledge basis, where the architecture to be evaluated is described;
- It must provide a form of a script/query language to enable transverse the architecture through their elements and respective relationships.

So, the EAMS[1], BiZZdesign Architect[2], ARIS ArchiMate Modeler[3], and IBM Rational System Architect[4] represent some of the existing EAM tools that endorse the requirements previously required. In this research, the EAMS is the EAM tool selected to be extended to enable the automatic compliance analysis.

Before proceeding to the proposed analysis it's also important to notice the following considerations.

The ArchiMate elements used to formalize the principles impact are presented in Table 14 (Appendix) as symbols. These symbols are based on (Šaša and Krisper, 2011). However, as can be seen not every elements belong to the ArchiMate metamodel. The new elements and relationships correspond to extensions to the metamodel (The Open Group, 2012). The reason for each extension is explained in the principle analysis that requires it.

It's also important to understand how the principle perspectives are defined. If PIA represents a set of all the elements and relationships of an EAD, then a viewpoint can be defined as a function vp that maps a given EA into a subset of its elements and their relations between them:

$$vp(PIA) = P, P \subseteq PIA,$$

[1] http://www.link.pt/eams/DefaultEA.aspx?idl=2
[2] http://www.bizzdesign.com/tools/bizzdesign-architect/
[3] http://www.softwareag.com/corporate/products/az/aris/default.asp
[4] http://www-03.ibm.com/software/products/en/ratisystarch

where P represents a view of the EA from the viewpoint vp. Hereupon, two functions are defined to represent a viewpoint (Šaša and Krisper, 2011):

- Function $Elt(x)$, where $x \subseteq PIA$, is a function which returns all elements in a given EAD x or in a given view x of an EA.
- Function $Rel(x)$, where$x \subseteq PIA$, is a function which returns all relationships in a given EAD x or in a given view x of an EA.

Finally, for each analysis some application examples will be illustrated. It's important to refer that the architecture elements recognized as green in the examples are recognized as compliant. On the opposite side, the elements marked in red are not recognized by the set applied and for that reason, they are considered non-compliant.

3.2 Architecture Principles Used

As mentioned in Section 2.2, the architecture principles choose is a critical part of this research. In Table 4, are presented the principles sufficiently objective to be evaluated through their architecture impact and for that reason are selected to perform this research. The principles identification in Table 4 is equal to their identification in (Greefhorst and Proper, 2011). Hereupon, the principles considered in this research are assigned in green.

A.1 ✗	A.2 ✗	A.3 ✗	A.4 ✗	A.5 ✗	A.6 ✗	A.7 ✗	A.8 ✗	A.9 ✗	A.10 ✗
A.11 ✓	A.12 ✓	A.13 ✗	A.14 ✓	A.15 ✓	A.16 ✗	A.17 ✗	A.18 ✗	A.19 ✗	A.20 ✗
A.21 ✗	A.22 ✗	A.23 ✗	A.24 ✗	A.25 ✗	A.26 ✗	A.27 ✗	A.28 ✓	A.29 ✓	A.30 ✗
A.31 ✗	A.32 ✗	A.33 ✗	A.34 ✗	A.35 ✗	A.36 ✗	A.37 ✗	A.38 ✗	A.39 ✗	A.40 ✗
A.41 ✗	A.42 ✓	A.43 ✓	A.44 ✗	A.45 ✗	A.46 ✗	A.47 ✗	A.48 ✗	A.49 ✗	A.50 ✗
A.51 ✗	A.52 ✗	A.53 ✗	A.54 ✗	A.55 ✗	A.56 ✓	A.57 ✗	A.58 ✗	A.59 ✗	

Table 4: The architecture principles considered in this research.

3.3 A.11 Principle Analysis

A.11 Front-Office Processes Are Separated from Back-Office Processes (FOSBO)
Type of information: business, data, application **Quality attributes:** maintainability
Rationale: • Front-office processes are different from back-office processes: the first is focused on customer intimacy while the other is focused on operational excellence. • Front-office processes require different skills and knowledge than back-office processes. • Separating back-office processes from front-office processes allows for reusing these back-office processes.
Implications: • Processes are dedicated to the front-office or back-office. • Disengagement between front-office and back-office processes is clearly defined.

• Front-office applications do not contain back-office logic.

Table 5: A.11 architecture principle adopted from (Greefhorst and Proper, 2011)

The compliance architecture analysis presented in this section is based on the principle highlighted above (Table 5). Concerning this analysis, it was necessary to proceed to a few extensions in the ArchiMate metamodel. These extensions are represented as follows:

- The sets **FOBP** and **BOBP** identify unambiguously what business processes correspond to front-office or back-office, respectively.
- The **FOL** and **BOL** enable to recognize the application components that implement front-office logic and back-office logic, respectively.

Hereupon, next is presented the A.11 principle compliance analysis.

3.3.1 Definition of FOSBO Perspectives

The FOSBO viewpoint (FOSBOV) addresses the elements and relationships of the set $FOSBOV \subseteq PIA$, which is defined with the following:

$$(1)\ Elt(FOSBOV) = \{x | (x \in BP) \vee (x \in AS, \exists bp \in BP: (x, bp) \in Used\ by \wedge \exists ac \in AC: (ac, x) \in Realization) \vee (x \in AC, \exists as \in AS: (x, as) \in Realization \wedge \exists bp \in BP: (as, bp) \in Used\ by)\}$$

$$(2) Rel(FOSBOV) = \{(x, y, z) | x \in BP, y \in AS, z \in AC: (y, x) \in Used\ by \vee (z, y) \in Realization\}$$

An illustration of the elements and relationships defined in the previous functions is presented in Figure 5.

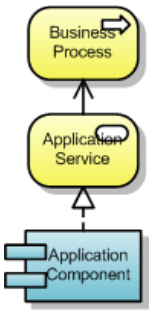

Figure 5: FOSBO viewpoint.

3.3.2 Definition of FOSBO Characteristics

The principle main goal pretends to separate clearly front-office from back-office. To achieve this goal, the principle prescribes that business processes only should be dedicated to front-office or back-office. This prescription is endorsed by the FOBPDed and BOBPDed sets, where are identified the processes dedicated to front and back office, respectively.

$$(3) FOBPDed = FOBP \setminus (FOBP \cap BOBP)$$

$$(4) BOBPDed = BOBP \backslash (FOBP \cap BOBP)$$

In Figure 6, an example of FOBPDed application is illustrated. The Figure 7 also highlights a BOBPDed application example.

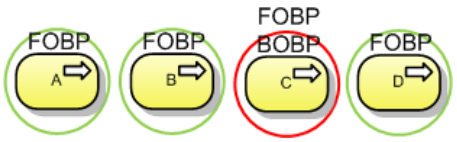

Figure 6 : FOBPDed application example.

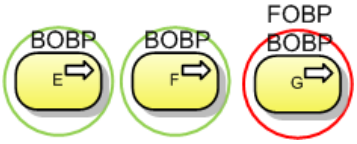

Figure 7 : BOBPDed application example.

The separation intended by the principle also impacts how business processes should be supported. It's prescribed that front-office business processes should be supported by application components that only implement front-office logic. The front-office business processes that address this prescription are defined by the FOBPAC set.

$$(5) FOBPAC = \{bp | bp \in FOBPDed \wedge (ACBP(bp) \subseteq ACFOL)\}$$

$$(6) ACFOL = FOL \backslash (FOL \cap BOL)$$

$$(7) ACBP(bp) = \{ac \mid ac \in AC \wedge ((\exists as \in AS: (ac, as) \in Realization \wedge (as, bp) \in Used\ by) \vee (ac, bp) \in Used\ by)\}$$

The ACFOL set identifies the application components that only implement front-office logic. The ACBP function enables to identify the application components that support a certain business process. Hereupon, it's shown in Figure 8 a FOBPAC application example.

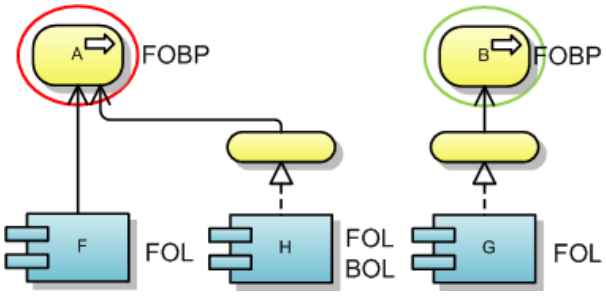

Figure 8 : FOBPAC application example.

Regarding the back-office business processes, it's only necessary to ensure that they are supported by application components with back-office logic implemented. These business processes are identified by the BOBPAC set.

$$(8) BOBPAC = \{bp | bp \in BOBPDed \wedge (ACBP(bp) \subseteq BOL)\}$$

A BOBPAC application example is highlighted in Figure 9.

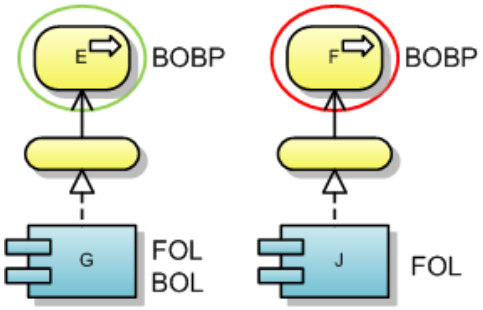

Figure 9 : BOBPAC application example.

The business processes that are not supported by application components could also be considered compliant, since they are dedicated to front or back office. These business processes are identified by the NSACBP set. An NSACBP example is illustrated in Figure 10.

$$(9) NSACBP = \{bp | bp \in (FOBPDed \cup BOBPDed) \wedge (\nexists a \in (AS \cup AC): (a, bp) \in Used\ by)\}$$

Figure 10 : NSACBP application example.

Hereupon, the FOSBO set defines the front-office and back-office business processes that are properly separated based on the principle prescriptions. The Figure 11 illustrates the application example of the FOSBO set based on the sequence of examples showed above.

$$(10) FOSBO = NSACBP \cup FOBPAC \cup BOBPAC$$

Figure 11 : FOSBO application example.

3.4 A.12 Principle Analysis

A.12 Channel-Specific Is Separated from Channel-Independent (CSSCI)
Type of information: business, data, application **Quality attributes:** reliability, efficiency, maintainability, portability
Rationale: • A lot of business activity is independent of the channel (telephone, mail, Internet, office) through which customers are contacted, and can be shared for multiple channels. • Data are ideally available through all channels, which is only possible when the data are managed in channel-independent processes.
Implications: • The activities at the borders of an end-to-end business process are specific to a channel and communicate with other activities in a channel-independent format. • Applications have dedicated components for channel-specific processing that interface with components that provide channel-independent business logic and data.

Table 6: A.12 architecture principle adopted from (Greefhorst and Proper, 2011).

The compliance architecture analysis presented in this section is based on the principle highlighted above (Table 6). Concerning this analysis, it was necessary to proceed to a few extensions in the ArchiMate metamodel. These extensions are represented as follows:

- The **BPIC** and **BPDC** sets identify the channel dependent and independent business processes, respectively.
- The **BII** and **BIT** sets recognize the business interfaces that correspond to Internet and Telephone channels. The channels used in this analysis could vary from organization to organization and for that reason other channels could be considered. However, in this analysis we consider the two channels previously referred.
- The **ACBL** set enables to identify the application components that implement business logic.
- The AC**CP** set identifies the application components dedicated to the channel processing.
- The **DODC** and **DOIC** sets recognize the data objects dependent and independent from channel, respectively.
- The **BODC** and **BOIC** sets identify the channel dependent and independent business objects, respectively.

Hereupon, next is presented the A.12 principle compliance analysis.

3.4.1 Definition of CSSCI Perspectives

The CSSCI viewpoint (CSSCIV) addresses the elements and relationships of the set $CSSCIV \subseteq PIA$, which is defined with the following:

$$(11) Elt(CSSCIV) = \{x | (x \in BP) \vee (x \in BO, \exists bp \in BP: (bp, x) \in Access \vee (x, bp) \in Access) \vee (x \in AS, \exists as \in AS: (as, x) \in Aggregation \vee \exists do \in DO: (x, do) \in Access) \vee (x \in AC, \exists as \in AS: (x, as) \in Realization) \vee (x \in DO, \exists bo \in BO: (x, bo) \in Realization) \vee (x \in BS, \exists bp \in BP: (bp, bs) \in Realization) \vee (x \in BI, \exists bs \in BS: (bs, x) \in Assignment)\}$$

$$(12) Rel(CSSCIV) = \{(x, y, u, s) \mid x \in BP, y \in BO, u \in BS, s \in BI: (x, y) \in Access \vee (y, x) \in Access \vee (x, u) \in Realization \vee (u, s) \in Assignment\} \cup \{(z, t, o, v, w) | z \in AC, t, o \in AS, v \in DO, w \in BO: (z, t) \in Realization \vee (t, o) \in Aggregation \vee (o, v) \in Access \vee (v, w) \in Realization)\}$$

An illustration of the elements and relationships defined in the previous functions is presented in Figure 12.

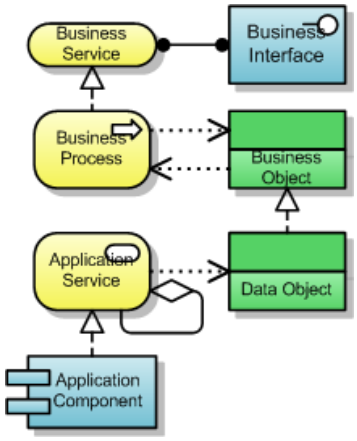

Figure 12: CSSVI viewpoint.

3.4.2 Definition of CSSCI Characteristics

Through the principle interpretation three main prescriptions have to be considered in this analysis.

The first one pretends to verify the consistency, that channel dependent business processes only could provide their business services through a unique channel. In the opposite case, the channel independent business processes can provide their business processes through several channels. The set that enables to identify the business processes that address this last prescription, is defined as follows.

$$(13)\ BPC = DCBPUC \cup BPIC$$

It's important to notice that DCBPUC set defines the prescription referred above about the dependent business processes. Relatively, to the channel independent business processes, there are no restrictions to be addressed and for reason they are compliant with this prescription.

$$(14) DCBPUC = \{bp \mid bp \in BPDC \wedge ((BusI(bp) \subseteq BII \wedge BusI(bp) \not\subset BIT) \vee (BusI(bp) \not\subset BII \vee BusI(bp) \subseteq BIT))\}$$

The BusI function used by the previous set, identifies the business interfaces from which the business services of a certain business process are provided.

$$(15) BusI(bp) = \{bi | bi \in BI \wedge (\exists bs \in BS: (bi, bs) \in Assignment \wedge (bp, bs) \in Realization)\}$$

The Figure 13 illustrates a simple example of the business processes identified by BPC.

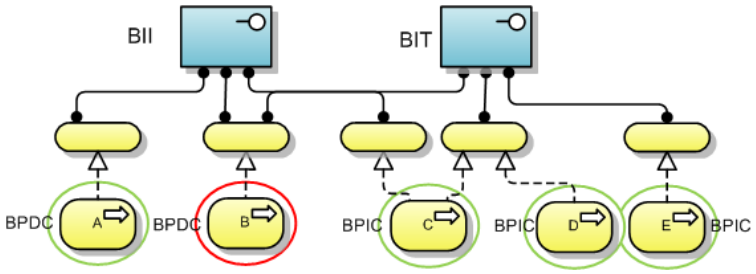

Figure 13: BPC application example.

Related with the last principle implication, it's perceptible that has to exist an application component only dedicated for the channel processing. This component acts as a middleware that interface application components that provide channel independent business logic and data. So, the interfaced application services have the restriction of not providing channel dependent data. The ASTCP set addresses the previous prescription, identifying the business logic application services interfaced by the channel processor and which do not provide dependent channel data. It's important to notice that the ASTCP formalization relatively to the channel processor middleware is based on (Van den Berg et al., 2007).

$$(16) ASTCP = \{as1 | as1 \in ASBL \wedge \exists as2 \in ASCP: (as2, as1) \in Aggregation \wedge \nexists do \in DODC: (as1, do) \in Access\}$$

The set ASCP set used in the ASTCP definition recognizes the application services provided by the channel processor components. The ASBL set identify the application services realized by application components that implement business logic.

$$(17) ASCP = \{as | as \in AS \wedge \exists ac \in (ACCP \backslash (ACCP \cap ACBL)): (ac, as) \in Realization\}$$

$$(18) ASBL = \{as | as \in AS \wedge \exists ac \in ACBL: (ac, as) \in Realization\}$$

The Figure 14 illustrates an ASTCP application example.

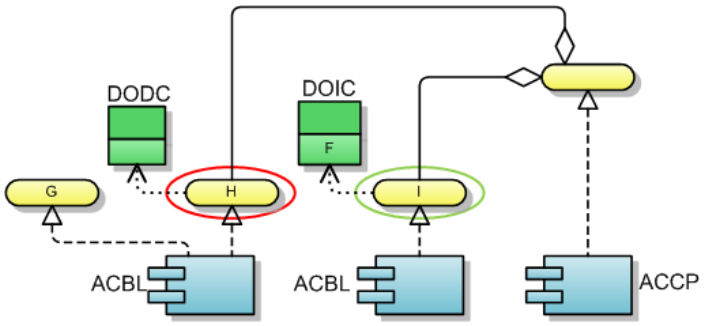

Figure 14 : ASTCP application example.

The principle also prescribes about the communication between business processes which could be inferred by the first principle implication. This communication can be seen as the information that passes between business processes.

It's prescribed that channel dependent business processes communicate with independent business processes with information in channel independent format. Other prescriptions that are not explicit in the principle concerning the communication between processes are also identified. These prescriptions don't conflict with the principle and for that reason, are also considered in this analysis. They prescribe that dependent business processes could also communicate with other dependent business processes in a dependent format, with the condition that any independent business processes don´t access to that information. The dependent processes could also communicate between them through independent channel information.

It's also important to notice that the channel independent information used in the referred communications should be result from application services provided through the channel processor component.

Hereupon, the BPDCDepC set identifies the channel dependent business processes that obey to the prescriptions previously mentioned.

$$(19)\ BPDCDepC = \{bp1 | bp1 \in BPDC \wedge ((\exists bo1 \in BODC: (bp1, bo1) \in Access \wedge \exists bp2 \in BPDC: (bo1, bp2) \in Access \wedge \nexists bp3 \in BPIC: (bo1, bp3) \in Access) \vee (\exists bo2 \in BOICDOT: (bp1, bo2) \in Access \wedge \exists bp3 \in (BPDC \cup BPIC): (bo2, bp3) \in Access))\}$$

The BOICDOT set used in the BPDCDepC, enables to recognize the channel independent business objects that result from data objects provided by application services interfaced by channel processors components. These data objects are channel independent objects.

$$(20)\ BOICDOT = \{bo | bo \in BOIC \wedge \exists do \in DOIC: (do, bo) \in Realization \wedge \exists as \in ASTCP: (as, do) \in Access\}$$

In Figure 15 is illustrated the obtained results from BOICDOT application.

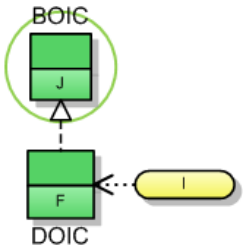

Figure 15: BOICDOT application example.

The Figure 16 illustrates a simple example of the BPDCDepC application.

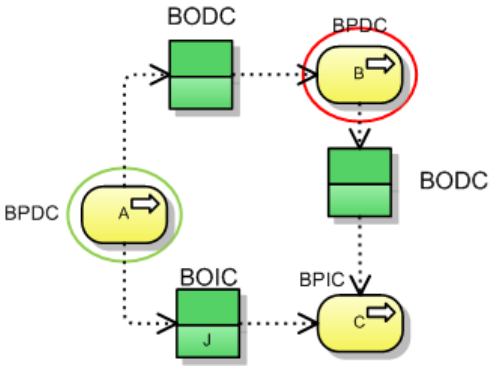

Figure 16: BPDCDepC application example.

Through the principle analysis, the independent business processes only could communicate in an independent channel format. Hereupon, the referred business process only could provide channel independent business objects, independently of the channel dependence that characterizes the business process that receive them. The business processes that address these prescriptions are defined in the following set.

$$(21) BPDCIndepC = \{bp1 | bp1 \in BPIC \wedge \exists bo \in BOIC: (bp1, bo) \in Access \wedge (\exists bp2 \in (BPDC \cup BPIC): (bo, bp2) \in Access\}$$

The Figure 17 presents an example of the BPDCIndepC application.

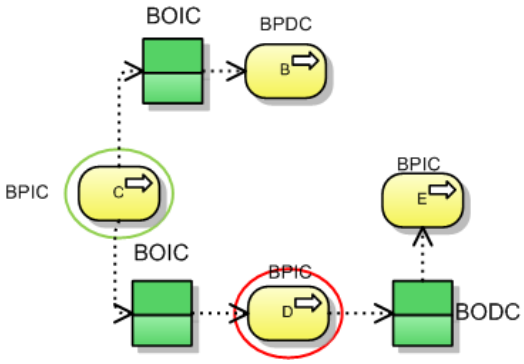

Figure 17: BPDCIndepC application example.

Hereupon, the business processes that address all the prescriptions related with the communication between processes are identified by the BPCom set.

$$(22) BPCom = BPDCDepC \cup BPDCIndepC$$

Finally, the CSSCI set endorses all principle prescriptions. This set identifies the compliant application services and business processes. A CSSCI application example is presented in Figure 18. The presented result is aligned with the sequence of the previous examples.

$$(23) CSSCI = (BPC \cap BPCom) \cup ASTCP$$

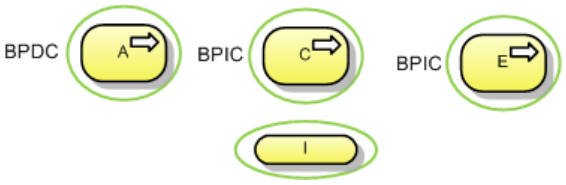

Figure 18 : CSSCI application example.

3.5 A.14 Principle Analysis

A.14 Data Are Provided by the Source (DPS)
Type of information: data, application **Quality attributes:** reliability, efficiency
Rationale: • When those who have the data also provide them, unnecessary intermediate layers (e.g. people or IT components) are prevented. • The performance and reliability of the data also increases, since each link in the chain adds performance overhead and potential errors.
Implications: • Electronic forms are provided to customers to enter their requests. • Applications acquire data from the source application.

Table 7: A.14 architecture principle adopted from (Greefhorst and Proper, 2011).

The compliance architecture analysis presented in this section is based on the principle highlighted above (Table 7). Concerning this analysis, it was necessary to proceed to a few extensions in the ArchiMate metamodel. These extensions are represented as follows:

- The **Creation and Provide relationships**. These relationships could be considered as a specification of the access relationship in the ArchiMate metamodel. The access relationship could be used to represent operations as create, provide, read or delete over a data object. Hereupon, to avoid ambiguity in the principle verification we need to extend the metamodel with the referred relationships.
- The **CR** set enables unambiguously identify the business roles that correspond to customers.
- The **EF** set is used to identify business interfaces that represent electronic forms.

Hereupon, next is presented the A.14 principle compliance analysis.

3.5.1 Definition of DPS Perspectives

The DPS viewpoint (DPSV) addresses the elements and relationships of the set $DPSV \subseteq PIA$, which is defined with the following:

$$(24) Elt(DPSV) = \{x | (x \in AS) \vee (x \in AC, \exists as1 : (x, as1) \in Realization) \vee (x \in DO, \exists as2 \in AS : (as2, x) \in Provide \vee (as2, x) \in Creation)\} \cup \{y | (y \in BS) \vee (y \in BI, \exists bs \in BS : (x, bs) \in Assignment \wedge \exists br \in BR : (y, br) \in Used\ by)\}$$

$$(25) Rel(DPSV) = \{(x, y, z) | x \in AS, y \in DO, z \in AC, ((x, y) \in Provide \vee (x, y) \in Creation) \wedge (z, x) \in Realization\} \cup \{(t, u, v) | t \in BR, u \in BI, v \in BS, (u, z) \in Assignment \wedge (u, t) \in Used\ by\}$$

An illustration of the elements and relationships defined in the previous functions is presented in Figure 19.

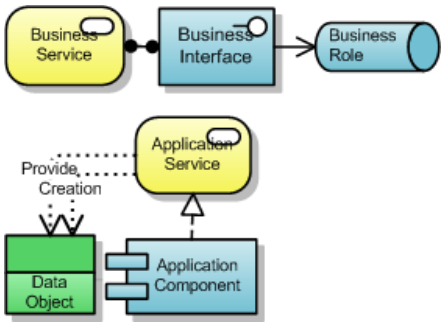

Figure 19: DPS viewpoint.

3.5.2 Definition of DPS Characteristics

By analyzing the referred principle, we perceive that we have to consider two main prescriptions. The first one is related with the fact that data objects only could be provided by application services from their application component source. The application source of a certain data object is the application component responsible for its creation.

This prescription is endorsed by the ASDOASource set, which defines the application services that only provide data objects if the application component responsible for those services, corresponds to the application source of the referred data objects.

$$(26) ASDOASource = \{as1 | as1 \in AS \wedge \forall do \in DO: (as1, do) \in Provide \wedge (\exists as2 \in ASAC(as1): (as2, do) \in Creation \vee (\exists ac \in AC: (ac, as1) \in Realization \wedge (ac, do) \in Creation))\}$$

The used ASAC function identifies all application services realized by the application component that realizes a determined application component. This function in ASDOASource is used to verify if any of the services identified by ASAC, is responsible for the creation of the data object being analyzed.

$$(27) ASAC(a) = \{as | as \in AS \wedge \exists ac \in AC: (ac, a) \in Realization \wedge (ac, as) \in Realization\}$$

The Figure 20 illustrates a simple example of the application services identified by the ASDOASource set.

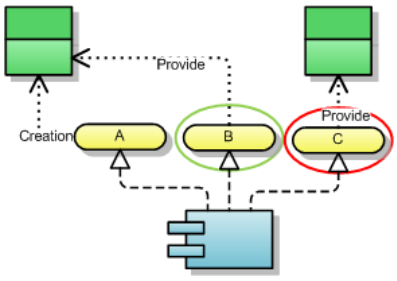

Figure 20: ASDOASource application example.

Through the first principle implication, it's also prescribed that should be provided electronic forms to customers enter their requests and use the respective business services. This prescription is endorsed by the BSEF set.

$$(28) BSElecF = \{bs | bs \in BS \wedge \exists bi \in EF: (bi, bs) \in Assignment \wedge \exists br \in CR: (bi, br) \in Used\ by\}$$

A simple example of BSElecF is presented in Figure 21.

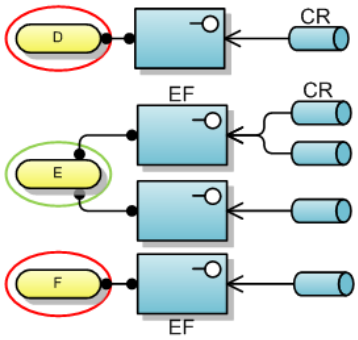

Figure 21: BSElecF application example.

Hereupon, the DPS set identifies the principle compliant elements. This set is composed by application services and business services. The application services are related with the prescription treated by ASDOASource and business services with the prescription considered in BSElecF.

$$(29) DPS = ASDOASource \cup BSElecF$$

In Figure 22 is presented a simple DPS application example. It's important to notice that the obtained result is in line with the sequence of the previous examples.

Figure 22: DPS application example.

3.6 A.15 Principle Analysis

A.15 Data Are Maintained in The Source Application (DMSA)
Type of information: data, application **Quality attributes:** reliability, efficiency, maintainability
Rationale: • Maintaining data in multiple places introduces risks of inconsistencies, which is undesirable at best. • It is inefficient to gather similar data from multiple places and resolve any potential conflicts.
Implications: • The source application for all types of data is known. • Applications acquire data from the source application. • Replication of data is accepted when properly motivated. • Replicas are never updated, unless a controlled synchronization mechanism is in place. • Data are not copied before it is finalized.

Table 8: A.15 architecture principle adopted from (Greefhorst and Proper, 2011).

The compliance architecture analysis presented in this section is based on the principle highlighted above (Table 8). Concerning this analysis, it was necessary to proceed to a few extensions in the ArchiMate metamodel. These extensions are represented as follows:

- The **Creation and Provide relationships**. These extensions are considered in the DPS compliance analysis and are also needed for this analysis. The explanation for this extension is presented in DPS analysis.
- The **RE** set defines the data objects considered as replicas.
- The **SM** set enables unambiguously identify the application components responsible for replica synchronization.

Hereupon, next is presented the A.15 principle compliance analysis.

3.6.1 Definition of DMSA Perspectives

The DMSA viewpoint (DMSAV) addresses the elements and relationships of the set $DMSAV \subseteq PIA$, which is defined with the following:

$$(31)\ Elt(DMSAV) = \{x | (x \in AS) \vee (x \in AC, \exists as1 \in AS: (x, as) \in Realization) \vee (x \in DO, \exists as2 \in AS: (as2, do) \in Provide \vee (as2, x) \in Creation \vee (as2, x) \in Access \vee (x, as2) \in Access)\}$$

$$(32) Rel(DMSAV) = \{(x, y, z) | x \in AS, y \in DO, z \in AC, ((x, y) \in Provide \vee (x, y) \in Creation \vee (x, y) \in Access \vee (y, x) \in Access) \wedge (z, x) \in Realization\}$$

An illustration of the elements and relationships defined in the previous functions is presented in Figure 23.

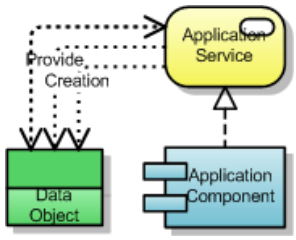

Figure 23: DMSA viewpoint.

3.6.2 Definition of DMSA Characteristics

The DMSA principle prescribes about who should provide a certain data object and about the replica synchronization.

Through the first and second implications, the principle prescribes that a data object only could be provided by application services from their application component source. This prescription is addressed by the ASDOASource set in the DPS analysis (Section 4.3.2), where are identified its compliant application services. An application example of this set is illustrated in Figure 20. This example will be used next to obtain the results of the DMSA compliant elements.

Regarding the replica synchronization, the principle prescribes that replica update only is performed if an application component responsible for the replica synchronization exists. If its existence is verified, the replica state only is updated by the application component responsible for that purpose. In the REUSM set are identified the replicas that address this last prescription.

$$(33)\ REUSM = \{do | do \in RE \wedge (\exists as1 \in SMAS: (as1, do) \in Access \vee \exists ac1 \in SM: (ac1, do) \in Access) \wedge (\nexists as2 \in (AS \backslash SMAS): (as2, do) \in Access \wedge \nexists ac2 \in (AC \backslash SM): (ac2, do) \in Access)\}$$

If the replica synchronization component existence is not verified, no application component can realize any operation in the existent replicas. Even when a synchronization mechanism exists and a replica is not updated it stills compliant if no other application read or access to it. So, these prescriptions are respected by the replicas identified in the RENUSM set.

$$(34) RENUSM = \{do | do \in RE \wedge (\nexists as1 \in SMAS: (as1, do) \in Access \wedge \nexists ac1 \in SM: (ac1, do) \in Access) \wedge (\nexists as2 \in (AS \backslash SMAS): (as2, do) \in Access \wedge (do, as2) \in Access \wedge \nexists ac2 \in (AC \backslash SM): (ac2, do) \in Access \wedge (do, ac2) \in Access)\}$$

Hereupon, the following set defines the replicas that address all prescriptions relatively to the replica synchronization.

$$(35)\ RESM = REUSM \cup RENUSM$$

The Figure 24 illustrates a simple example of RESM application.

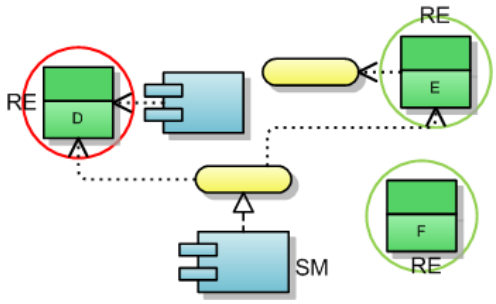

Figure 24: RESM application example.

The SMAS set used in the REUSM and RENUSM definition, identifies the application services realized by the application components responsible for replica synchronization.

$$(36) SMAS = \{as | as \in AS \wedge \exists ac \in SM : (ac, as) \in Realization\}$$

Hereupon, the DMSA set identifies the principle compliant elements. This set is composed by application services and data objects. The application services are related with the elements identified in ASDOASource set and the data objects in RESM.

$$(37) DMSA = ASDOASource \cup RESM$$

In Figure 25 is presented a simple DMSA application. It's important to notice that the obtained result is in line with the sequence of the previous illustrated examples (Figure 20 and Figure 24).

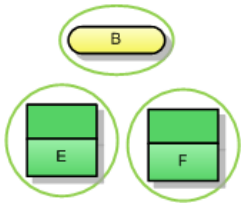

Figure 25: DMSA application example.

3.7 A.28 Principle Analysis

A.28 Applications Are Modular (AAM)
Type of information: application **Quality attributes:** reliability, maintainability, portability
Rationale: • Modularized applications are much easier to develop, maintain, reuse and migrate than monolithical applications. • Modularized applications are also more reliable since changes have a more localized and therefore predictable impact.
Implications: • Applications are decomposed into components that have limited and acyclical dependencies on other components. • Application components are units of configuration management and deployment. • Application components have a logical and documented layered structure, where lower level layers are independent of higher level layers. • Presentation logic, process logic, business logic and data exist in separate layers or components.

Table 9: A.28 architecture principle adopted from (Greefhorst and Proper, 2011).

The compliance architecture analysis presented in this section is based on the principle highlighted above (Table 9). Concerning this analysis, it was necessary to proceed to a few extensions in the ArchiMate metamodel. These extensions are represented as follows:

- The **ACPreL**, **ACProL**, **ACBL** and **ACDL** sets enable to identify the application components that implement presentation, process, business and data logic, respectively.

Hereupon, next is presented the A.28 principle compliance analysis.

3.7.1 Definition of AAM Perspectives

The AAM viewpoint (AAMV) addresses the elements and relationships of the set $AAMV \subseteq PIA$, which is defined with the following:

$$(38) Elt(AAMV) = \{x | (x \in AC) \vee (x \in AS, \exists ac1, ac2 \in AC : (ac1, x) \in Realization \wedge (x, ac2) \in Used\ by)\}$$

$$(39) Rel(AAMV) = \{(x, y) \,|\, x, y \in AC : (x, y) \in Composition \vee (x, y) \in Aggregation\} \cup \{(t, z) | z \in AC, t \in AS : (z, t) \in Realization \vee (t, z) \in Used\ by\}$$

An illustration of the elements and relationships defined in the previous functions is presented in Figure 26.

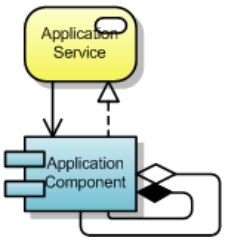

Figure 26: AAM viewpoint.

3.7.2 Definition of AAM Characteristics

Concerning the AAM analysis, the principle prescribes that each monolithical application component only should implement one type of logic. The MoAC1Log set identifies the applications components that endorse this prescription.

$$(40) MoAC1Log = MoAC \backslash (MoAC \cap ACSevLog)$$

The MoAC set identifies the monolithical application components and ACSevLog defines the application components that implement at least two types of logic.

$$(41) MoAC = \{ac1 | ac1 \in AC \wedge (\nexists ac2 \in AC : (ac1, ac2) \in Composition \vee (ac1, ac2) \in Aggregation)\}$$

$$(42) ACSevLog = (ACPreL \cap ACProL) \cup (ACPreL \cap ACBL) \cup (ACPreL \cap ACDL) \cup (ACProL \cap ACBL) \cup (ACProL \cap ACDL) \cup (ACBL \cap ACDL)$$

In Figure 27 is illustrated an example of the MoAC1Log application.

Figure 27: MoAC1Log application example

The principle also prescribes about the dependency between application components. This dependency is related to the usage that a component makes from other component. To control this dependency, the application components that correspond to a layer with lower abstraction level should be independent of higher level components. So, the components with a certain type of logic only could use application components with lower or equal abstraction level logic. To verify this prescription, the different logics have to be classified depending on the abstraction level. From the higher level to the lower, the presentation logic is followed by the process, business and data logic. The ACMUsage set identifies the monolithical components that address the dependency prescription. The ACPreLMUsage, ACProLMUsage, ACBLMUsage and ACDLMUsage sets define for each logic, the components that endorse the dependency prescription. The independent application components are also compliant with the dependency prescription, which are recognized by the ACMUsageInd set.

$$(43) ACMUsage = ACPreLMUsage \cup ACProLMUsage \cup ACBLMUsage \cup ACDLMUsage \cup ACMUsageInd$$

$$(44)ACPreLMUsage = \{ac1|ac1 \in (ACPreL \cap MoAC1Log) \wedge (\exists ac2 \in MoAC1Log: (ac2, ac1) \in Used\ by \vee (\exists as1 \in AS: (as1, ac1) \in Used\ by \wedge (ac2, as1) \in Realization)) \wedge (\nexists ac3 \in AC \backslash MoAC1Log: (ac3, ac1) \in Used\ by \wedge (\nexists as2 \in AS: (as2, ac1) \in Used\ by \wedge (ac3, as2) \in Realization))\}$$

$$(45)ACProLMUsage = \{ac1|ac1 \in (ACProL \cap MoAC1Log) \wedge (\exists ac2 \in (ACProL \cup ACBL \cup ACDL) \cap MoAC1Log: (ac2, ac1) \in Used\ by \vee (\exists as1 \in AS: (as1, ac1) \in Used\ by \wedge (ac2, as1) \in Realization)) \wedge (\nexists ac3 \in AC \backslash ((ACProL \cup ACBL \cup ACDL) \cap MoAC1Log): (ac3, ac1) \in Used\ by \wedge (\nexists as2 \in AS: (as2, ac1) \in Used\ by \wedge (ac3, as2) \in Realization))\}$$

$$(46)ACBLMUsage = \{ac1|ac1 \in (ACBL \cap MoAC1Log) \wedge (\exists ac2 \in (ACBL \cup ACDL) \cap MoAC1Log: (ac2, ac1) \in Used\ by \vee (\exists as1 \in AS: (as1, ac1) \in Us1ed\ by \wedge (ac2, as1) \in Realization)) \wedge (\nexists ac3 \in AC \backslash ((ACBL \cup ACDL) \cap MoAC1Log): (ac3, ac1) \in Used\ by \wedge (\nexists as2 \in AS: (as2, ac1) \in Used\ by \wedge (ac3, as2) \in Realization))\}$$

$$(47)ACDLMUsage = \{ac1|ac1 \in (ACDL \cap MoAC1Log) \wedge (\exists ac2 \in (ACDL \cap MoAC1Log): (ac2, ac1) \in Used\ by \vee (\exists as1 \in AS: (as1, ac1) \in Used\ by \wedge (ac2, as1) \in Realization)) \wedge (\nexists ac3 \in AC \backslash (ACDL \cap MoAC1Log): (ac3, ac1) \in Used\ by \wedge (\nexists as2 \in AS: (as2, ac1) \in Used\ by \wedge (ac3, as2) \in Realization))\}$$

$$(48)ACMUsageInd = \{ac1|ac1 \in MoAC1Log \wedge (\nexists ac2 \in AC: (ac2, ac1)) \in Used\ by) \wedge (\nexists as1 \in AS: (as1, ac1) \in Used\ by)\}$$

The Figure 28 illustrates a simple example of the application components identified by the ACMUsage application.

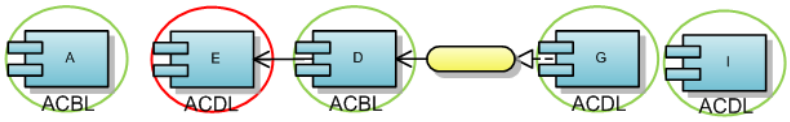

Figure 28: ACMUsage application example.

The principle also prescribes about the application component composition. The application components only should be composed by application components in accordance with the prescriptions previously analyzed. This guarantees the application layered structure intended by the principle. So, the ACCompACM1Log set identifies the components that respect this prescription.

$$(48)ACCompACM1Log = \{ac|ac \in (AC \backslash MoAC) \wedge (AppComp(ac) \subseteq ACMUsage)\}$$

It's relevant to notice that AppComp function identifies the application components that composes a determined application component.

$$(49)AppComp(x) = \{ac|ac \in AC \wedge \big((x, ac) \in Composition \vee (x, ac) \in Aggregation\big)\}$$

The Figure 29 illustrates a simple example of ACCompACM1Log set.

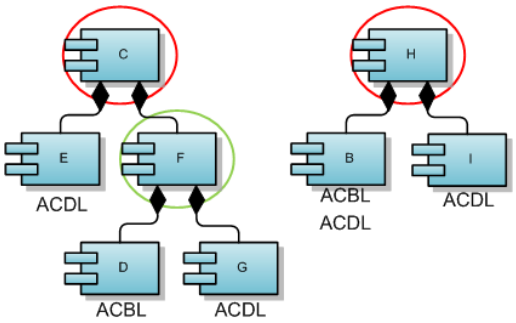

Figure 29: ACCompACM1Log application example.

Hereupon, the principle compliant application components are identified by the AAM set. In Figure 30 is presented the obtained result of the referred set. It's important to notice that this result is in line with the sequence of the previous examples.

$$(50) AAM = ACMUsage \cup ACCompACM1Log$$

Figure 30: AAM application example.

3.8 A.29 Principle Analysis

A.29 Application Functionality is Available Through an Enterprise Portal (AFTEP)
Type of information: application **Quality attributes:** usability
Rationale: • A portal provides functionality that is targeted at the role and personal preferences of the user, optimally supporting users in their work. • A portal provides a single point of access, and integration of functionality at the glass, relieving users from manually finding and integrating functionality. • A portal can provide single sign-on to users.
Implications: • There is an Enterprise Portal that provides access to all application functionality. • All applications are portal-enabled, exposing their functionality as portlets/web parts.

Table 10: A.29 architecture principle adopted from (Greefhorst and Proper, 2011).

The compliance architecture analysis presented in this section is based on the principle highlighted above (Table 10). Concerning this analysis, it was necessary to proceed to a few extensions in the ArchiMate metamodel. These extensions are represented as follows:

- The **EP** set identifies the application components that correspond to an enterprise portal.
- The **WP** set identifies the application interfaces that are considered web parts.

Hereupon, next is presented the A.29 principle compliance analysis.

3.8.1 Definition of AFTEP Perspectives

The AFTEP viewpoint (AFTEPV) addresses the elements and relationships of the set $AFTEPV \subseteq PIA$, which is defined with the following:

$$(51) Elt(AFTEPV) = \{x | (x \in AS) \vee (x \in AC, \exists as \in AS: (ac, as) \in Realization) \vee (x \in AI, \exists as \in AS: (x, as) \in Assignment)\}$$

$$(52)\ Rel(AFTEPV) = \{(x, y, t, z) \,|\, x, t \in AS, y \in AC, z \in AI: (y, x) \in Realization \vee (z, t) \in Assignment \vee (x, t) \in Aggregation\}$$

An illustration of the elements and relationships defined in the previous functions is presented in Figure 31.

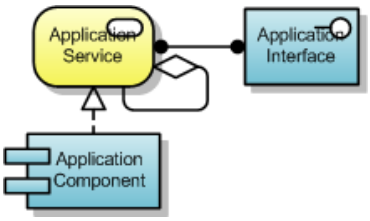

Figure 31: AFTEP viewpoint.

3.8.2 Definition of AFTEP Characteristics

In the AFTEP compliance analysis, two prescriptions have to be verified. The first one is related with the first implication, which prescribes that all application services have to be provided through an enterprise portal. This enterprise portal acts as a unique point of access, which is responsible to interface the access to all application services. This aspect is defined in the ASTEP set, where are identified the application services that address this prescription. The ASTEP definition is based on (Van den Berg et al., 2007).

$$(53) ASTEP = \{as1 \mid as1 \in (AS \backslash ASEP) \wedge \exists as2 \in ASEP : (as2, as1) \in Aggregation\}$$

It's important to notice that ASEP set defines the application services realized by the enterprise portal. These services are responsible to present all services through the portal.

$$(54) ASEP = \{as | as \in AS \wedge \exists ac \in EP : (ac, as) \in Realization\}$$

The Figure 32 illustrates a simple example of the application services identified by ASTEP.

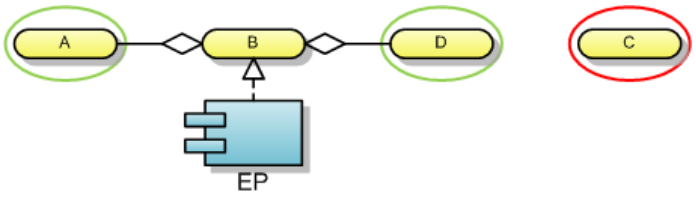

Figure 32: ASTEP application example.

The other prescription to be verified corresponds to the last principle implication. It's prescribed that all application services, including the application services realized by the enterprise portal have to be exposed as Web Parts. So, the ASPWP set defines all application services provided as Web Part.

$$(55) ASPWP = \{as | as \in AS \wedge \exists ai \in WP : (ai, as) \in Assignment\}$$

In Figure 33 is illustrated a simple application of the set defined above.

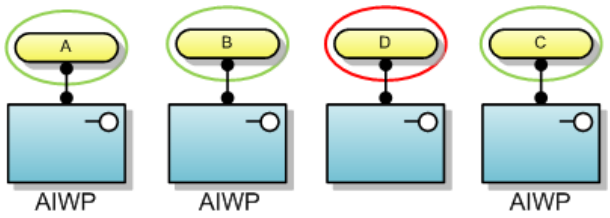

Figure 33: ASPWP application example.

Hereupon, the AFTEP set aggregates the prescriptions previously mentioned, where are identified the principle compliant application services. The application of this set is presented in Figure 34, which is in line with the sequence of the previous examples.

$$(56) AFTEP = (ASTEP \cup ASEP) \cap ASPWP$$

Figure 34: AFTEP application example.

3.9 A.42 Principle Analysis

A.42 Presentation Logic, Process Logic and Business Logic Are Separated (PPBLS)
Type of information: application **Quality attributes:** maintainability
Rationale: • These forms of logic are inherently different, and it should be possible to change them independently. • By separating these forms of logic they can be reused independently from each other.
Implications: • Presentation logic, process logic and business logic are implemented in separate application components. • Components have a layered dependency structure, with minimal dependencies. • Data are only managed in components that implement the business logic.

Table 11: A.42 architecture principle adopted from (Greefhorst and Proper, 2011).

The compliance architecture analysis presented in this section is based on the principle highlighted above (Table 11). Concerning this analysis, it was necessary to proceed to a few extensions in the ArchiMate metamodel. The extensions endorsed by this principle are represented by the ACPreL, ACBL and ACDL sets, which had been presented previously in Section 3.7. Hereupon, next is presented the A.42 principle compliance analysis.

3.9.1 Definition of PPBLS Perspectives

The PPBLS viewpoint (PPBLSV) addresses the elements and relationships of the set $PPBLV \subseteq PIA$, which is defined with the following:

$$(57) Elt(PPBLSV) = \{x | (x \in AC) \vee (x \in AS, \exists ac1, ac2 \in AC: (ac1, x) \in Realization \wedge (x, ac2) \in Used\ by) \vee (x \in DO, (\exists ac3 \in AC: (ac3, x) \in Access \vee (x, ac3) \in Access) \vee (\exists as \in AS: (as, x) \in Access \vee (x, as) \in Access))\}$$

$$(58) Rel(PPBLSV) = \{(x, y, t, w) | x, y \in AC, t \in AS, w \in DO: (x, y) \in Composition \vee (x.y) \in Aggregation \vee (x, y) \in Used\ by \vee (x, t) \in Realization \vee (t, x) \in Used\ by \vee (x, w) \in Access \vee (w, x) \in Access \vee (t, w) \in Access \vee (w, t) \in Access\}$$

An illustration of the elements and relationships defined in the previous functions is presented in Figure 35.

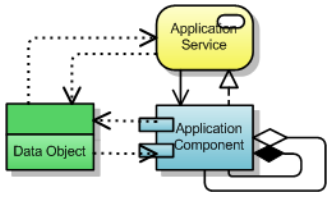

Figure 35: PPBLS viewpoint.

3.9.2 Definition of PPBLS Characteristics

This PPBLS principle presents several similar prescriptions with the AAM principle. The prescriptions about how many types of logic an application component could implement and about the dependency between application components are also considered in this analysis. However, the sets that endorse these prescriptions in the PPBLS analysis requires a slight reformulation of the sets defined in the AAM analysis.

The PPBLS principle prescribes that each monolithical application component only should implement one type of logic. The MoAC1L set identifies the application components that endorse this prescription.

$$(59) MoAC1L = MoAC \backslash (MoAC \cap ACSevL)$$

The set MoAC is defined in the AAM analysis and is responsible to identify the monolithical application components. The ACSevL identifies the application components that implement at least two types of logic.

$$(60) ACSevL = (ACPreL \cap ACBL) \cup (ACPreL \cap ACDL) \cup (ACBL \cap ACDL)$$

In Figure 36 is illustrated an example of the MoAC1L application.

Figure 36: MoAC1L application example.

As stated by the second principle implication, this principle pretends that application components achieve a layered dependency structure. The prescription about dependency between application components helps to achieve this goal. Hereupon, this prescription advises that application components with a certain abstraction level only should use application components with lower or equal abstraction level logic. To verify this prescription, it's needed to classify the different logics depending on the abstraction level that characterizes each one. From the higher value of abstraction to the lower, the presentation logic is the higher followed by business logic and finally, by data logic. The set ACMU identifies the monolithical application components that follow the prescription above. The ACPreLMU, ACBLMU and ACDLMU sets define for each logic, the components that endorse the dependency prescription. The independent application components are also compliant with the dependency prescription, which are recognized by the ACMUInd set.

$$(61) ACMU = ACPreLMU \cup ACBLMU \cup ACDLMU \cup ACMUInd$$

$$(62) ACPreLMU = \{ac1 | ac1 \in (ACPreL \cap MoAC1L) \wedge (\exists ac2 \in MoAC1L: (ac2, ac1) \in Used\ by \vee (\exists as1 \in AS: (as1, ac1) \in Used\ by \wedge (ac2, as1) \in Realization)) \wedge (\nexists ac3 \in AC \backslash MoAC1L: (ac3, ac1) \in Used\ by \wedge (\nexists as2 \in AS: (as2, ac1) \in Used\ by \wedge (ac3, as2) \in Realization))\}$$

$$(63) ACBLMU = \{ac1 | ac1 \in (ACBL \cap MoAC1L) \wedge (\exists ac2 \in (ACBL \cup ACDL) \cap MoAC1L: (ac2, ac1) \in Used\ by \vee (\exists as1 \in AS: (as1, ac1) \in Used\ by \wedge (ac2, as1) \in Realization)) \wedge (\nexists ac3 \in AC \backslash ((ACBL \cup ACDL) \cap MoAC1L): (ac3, ac1) \in Used\ by \wedge (\nexists as2 \in AS: (as2, ac1) \in Used\ by \wedge (ac3, as2) \in Realization))\}$$

$$(64) ACDLMU = \{ac1 | ac1 \in (ACDL \cap MoAC1L) \wedge (\exists ac2 \in (ACDL \cap MoAC1L): (ac2, ac1) \in Used\ by \vee (\exists as1 \in AS: (as1, ac1) \in Used\ by \wedge (ac2, as1) \in Realization)) \wedge (\nexists ac3 \in AC \backslash (ACDL \cap MoAC1L): (ac3, ac1) \in Used\ by \wedge (\nexists as2 \in AS: (as2, ac1) \in Used\ by \wedge (ac3, as2) \in Realization))\}$$

$$(65) ACMUInd = \{ac1 | ac1 \in MoAC1L \wedge (\nexists ac2 \in AC: (ac2, ac1)) \in Used\ by) \wedge (\nexists as1 \in AS: (as1, ac1) \in Used\ by)\}$$

An ACMU application example is illustrated in Figure 37.

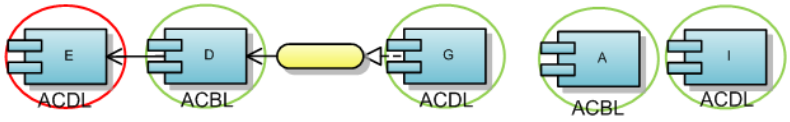

Figure 37: ACMU application example.

Regarding the last principle implication, it's prescribed that application components that implement business logic are the only that could manage data objects. In this analysis, the data objects management is characterized by operations that could or not influence the state of a certain data object. Hereupon, the business logic application components are the only that could alter a certain data object. However, all the operations that have no impact in the referred data object could be executed by any application component, regardless of the implemented logic. The ACMDO defines the application components that respect the data management object prescribed above. In this case, the business logic application components don't are restricted as the presentation and data logic, so relative to the data object management all monolithical business logic components are compliant.

$$(65) ACMDO = (ACBL \cap MoAC1L) \cup ACPreLMDO \cup ACDLMDO$$

$$(66) ACPreLMDO = \{ac | ac \in (ACPreL \cap MoAC1L) \wedge (\nexists do \in DO: (ac, do) \in Access \wedge (\nexists as \in ASAComp(ac): (as, do) \in Acces))\}$$

$$(67) ACDLMDO = \{ac | ac \in (ACDL \cap MoAC1L) \wedge (\nexists do \in DO: (ac, do) \in Access \wedge (\nexists as \in ASAComp(ac): (as, do) \in Acces))\}$$

$$(68) ASAComp(ac) = \{as | as \in AS: (ac, as) \in Realization\}$$

The ACPreLMDO and ACDLMDO identify the presentation and data application components that only perform operations that have no impact in any data object. The ASAComp function identifies the application services realized by a certain application component. In Figure 38 is showed an example with the identification of compliant application components with this prescription.

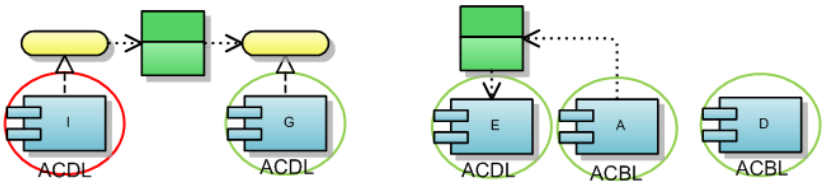

Figure 38: ACMDO application example.

Hereupon, all monolithical application components should respect the prescriptions previous mentioned. So, the ACMUDO identifies the application components that address the prescriptions related with the dependency between applications and the data object management. Based on the results of the previous examples, it's presented in Figure 39 the result of the ACMUDO set.

$$(69) ACMUDO = ACMU \cap ACMDO$$

Figure 39: ACMUDO application example.

Finally, this principle such as the AAM principle pretends that application components have a layered dependency structure. To achieve this goal, another prescription has to be addressed by the application components that are composed by other application components. These components have to be composed by application components that obey the dependency and data object management prescriptions. Hereupon, the ACCompACM1L set identify the application components composed by application components that address all prescriptions previously considered in this analysis.

$$(70) ACCompACM1L = \{ac \mid ac \in (AC \backslash MoAC) \wedge (AppComp(ac) \subseteq ACMUDO)\}$$

In Figure 40 is illustrated an ACCompACM1L application example.

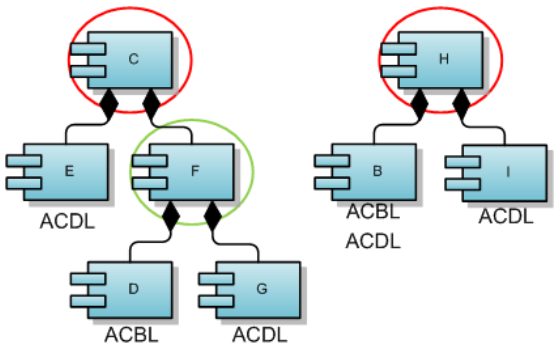

Figure 40:ACCompACM1L application example.

Hereupon, all prescriptions to be addressed in the PPBLS principle are defined in the PPBLS set. In Figure 41, based on the example sequence presented above, is presented the result of the PPBLS application.

$$(71) PPBLS = ACMUDO \cup ACCompACM1L$$

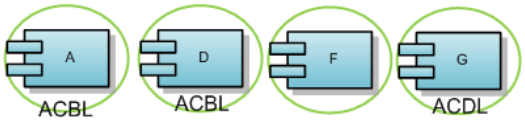

Figure 41: PPBLS application example.

3.10 A.43 Principle Analysis

A.43 IT Systems Communicate Through Services (ITSCTS)
Type of information: data, application, technology **Quality attributes:** efficiency, maintainability, portability
Rationale: • Services can be reused, which leads to less interfaces and is thus much more efficient. • By reusing services new solutions can be assembled much faster, resulting in a shorter time-to-market.
Implications: • Services are defined for all data and functionality that IT systems provide to other IT systems. • Services are defined as reusable as possible, shielding implementation details and adhering to interface standards, formats and protocols. • Services are published in a service directory where they can be found for reuse.

Table 12: A.43 architecture principle adopted from (Greefhorst and Proper, 2011).

The compliance architecture analysis presented in this section is based on the principle highlighted above (Table 12). Concerning this analysis, it was necessary to proceed to a few extensions in the ArchiMate metamodel. These extensions are represented as follows:

- The **AIS** and **IIS** sets enable to identify if an application interface and infrastructure interface correspond to a service, respectively.
- The **W3C** set identifies the application interfaces and infrastructure interfaces that are implemented based on the W3C protocol. To perform the proposed analysis, other attributes relative to other protocols could be used. The used protocols could vary from organization to organization, however in this particular analysis we will use the W3C protocol.
- The **SD** set identifies the system software that represents the service directory.
- The **PubS** and **SeaS** sets enable unambiguously to identify the infrastructure services that correspond to publish service and search service in a service directory, respectively.
- The **SDPubS** set identify the artifact that contains the information about the published services in the service directory.
- The **SerInf** set identifies the artifacts that correspond to the information published about a certain application or infrastructure service.

Hereupon, next is presented the A.43 principle compliance analysis.

3.10.1 Definition of ITSCTS Perspectives

The ITSCTS viewpoint (ITSCTSV) addresses the elements and relationships of the set $ITSCTSV \subseteq PIA$, which is defined with the following:

$$(72) Elt(ITSCTSV) = \{x | (x \in ART) \vee (x \in AS, \exists art \in ART: (x, art) \in Association) \vee (x \in IS, \exists art \in ART: (x, art) \in Association) \vee (x \in AI, \exists as \in AS: (x, as) \in Assignment) \vee (x \in II, \exists is \in IS: (x, is) \in Assignment) \vee (x \in SS, \exists is \in IS: (x, is) \in Realization)\}$$

$$(73) Rel(ITSCTSV) = \{(x, y, z, w, t, u, v) | x \in AI, y \in AS, z, w \in ART, t \in II, u \in IS, v \in SS: (x, y) \in Assignment \vee (y, z) \in Association \vee (z, w) \in Aggregation \vee (u, z) \in Association \vee (u, z) \in Access \vee (z, u) \in Access \vee (t, u) \in Assignment \vee (v, u) \in Realization\}$$

An illustration of the elements and relationships defined in the previous functions is presented in Figure 42.

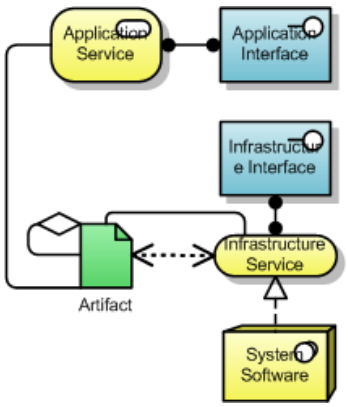

Figure 42: ITSCTS viewpoint.

3.10.2 Definition of ITSCTS Characteristics

Regarding the first and the second implications, the ITSCTS principle prescribes that application and infrastructure services should be provided through a service. It's also prescribed that formal protocols/standards should be used in the service definition. As mentioned in the previous section, the protocol used in this analysis is the W3C protocol. Hereupon, the ASISServP set identifies the application and infrastructure services provided through a service defined based on a formal protocol. The ASServP set identifies the application services and the ISServP identifies the infrastructure services.

$$(74) ASISServP = ASServP \cup ISServP$$

$$(75) ASServP = \{as | as \in AS \wedge \exists ai \in (AIS \cap W3C): (ai, as) \in Assignment\}$$

$$(76) ISServP = \{is | is \in IS \wedge \exists ii \in (IIS \cap W3C): (ii, is) \in Assignment\}$$

In Figure 43 an application example of the ASISServP set is illustrated.

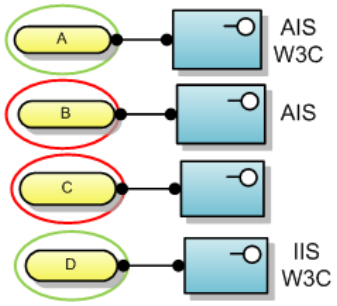

Figure 43: ASISServP application example.

If we want to extend this analysis to consider another protocol you simply have to do the analysis with the new protocol and then make the reunion between the sets.

Regarding the last principle implication, it prescribes that all application and infrastructure services should be published in a service directory. Hereupon, it's needed to evaluate if the information relative to each service is published in the service directory. This prescription is analyzed in the AISPubS set where are identified the application and infrastructure services published in a service directory.

$$(77) AISPubS = \{as \mid as \in AS \wedge (\exists art1 \in ServInf: (art1, as) \in Association \wedge \exists art2 \in SDInfPubS: (art2, art1) \in Aggregation)\} \cup \{is \mid is \in IS \wedge (\exists art3 \in ServInf: (art3, is) \in Association \wedge \exists art4 \in SDInfPubS: (art4, art3) \in Aggregation)\}$$

The SDInfPubS set used above identifies the artifacts that correspond to the service directories information, where is endorsed the information about the published services. The principle also prescribes, that these artifacts should be managed by service directories (system software) that provide services to publish and search for a service. It's important to notice that in this analysis, the publish service is the only service responsible to alter the information about the services published in the service directory.

$$(78) SDInfPubS = \{art \mid art \in SDPubS \wedge (\exists as1 \in PubS: (as1, art) \in Access \wedge \exists as2 \in SeaS: (art, as2) \in Access \wedge \exists ss \in SD: (ss, as1) \in Realization \wedge (ss, as2) \in Realization)\}$$

In Figure 44 is illustrated the artifacts recognized by the SDInfPubS set.

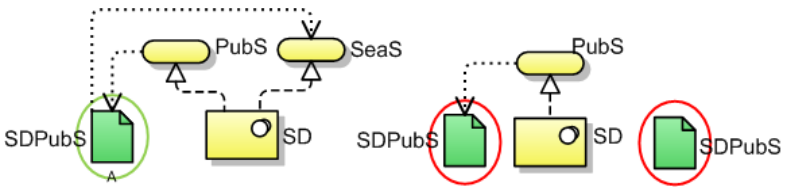

Figure 44: SDInfPubS application example.

An application of AISPubS set is also presented in Figure 45.

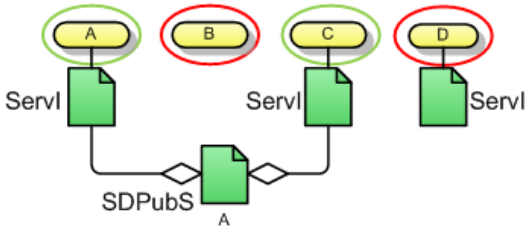

Figure 45: AISPubS application example.

Hereupon, the ITSCTS set identifies the compliant application and infrastructure services with this principle. These services address all the prescriptions previously mentioned.

$$(79) ITSCTS = ASISServP \cap AISPubS$$

In Figure 46 are shown the elements recognized by ITSCTS. It's important to notice, that the obtained results are in line with the sequence of examples previously presented.

Figure 46: ITSCTS application example.

3.11 A.56 Principle Analysis

A.56 Integration with External IT Systems Is Localized in Dedicated IT Components (IESLD)
Type of information: application, technology **Quality attributes:** functionality, maintainability
Rationale: • Using dedicated IT components for integration with external IT systems is more efficient and manageable since interface costs are spent only once, and changes can be limited to one component. • Dedicated IT components can provide a first line of defense for security attacks. • B2B integration is often more complex due to special interchange protocols, formats and agreements which require dedicated middleware.
Implications: • Applications contain IT components dedicated to integration in the business logic layer, which can be used from the presentation layer. • IT components are selected and used to support the interchange protocols, formats and agreements that are needed for integration with other organizations.

Table 13: A.56 architecture principle adopted from (Greefhorst and Proper, 2011).

The compliance architecture analysis presented in this section is based on the principle highlighted above (Table 13). Concerning this analysis, it was necessary to proceed to a few extensions in the ArchiMate metamodel. These extensions are represented as follows:

- The **ACPreL** and **ACBL** sets enable to identify the application components that implement presentation and business logic, respectively. These sets were also considered in A.28 and A.42 principles analysis.
- The **ACI** and **ACE** sets identify the internal and external application components to the organization, respectively. The need for these set,s is due to the principle impact in the integration between organizations.
- The **ACIntg** allows identifying the application components dedicated to the integration between internal and external application components.
- The **W3C** set endorses two types of application components. One is related with the integration application components that expose services supporting the W3C protocol. These application components are identified by the W3C set. The other application components identified are referent to the external components that requires that the used services are integrated based on the referred protocol. In the proposed analysis we consider the W3C protocol. It's also important to notice that other protocols could be used, because the used protocols depend from organization to organization.

Hereupon, next is presented the A.56 principle compliance analysis.

3.11.1 Definition of IESLD Perspectives

The IESLD viewpoint (IESLDV) addresses the elements and relationships of the set $IESLDV \subseteq PIA$, which is defined with the following:

$$(80) Elt(IESLD) = \{x | (x \in AS) \vee (x \in AC, \exists as \in AS: (ac, as) \in Realization \vee (as, ac) \in Used\ by)\}$$

$$(81) Rel(IESLD) = \{(x, y, t) | x, t \in AS, y \in AC: (y, x) \in Realization \vee (t, y) \in Used\ by\ \vee (x, t) \in Aggregation\}$$

An illustration of the elements and relationships defined in the previous functions is presented in Figure 47.

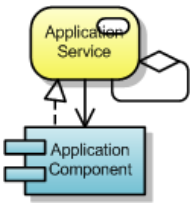

Figure 47: IESLD viewpoint.

3.11.2 Definition of IESLD Characteristics

The IESLD principle pretends to impact how the B2B integration is performed. Under this principle are perceptible three main prescriptions that should be respected. One prescribes the presence of a dedicated application component responsible for the referred integration. This dedicated component acts as a middleware that interface the application services used by external organizations. These services interfaced, through the integration component, should contain business logic. In other words, the application components that provide these services have to implement business logic. Hereupon, the ASBLIntg set defines the application services with business logic that are exposed through the integration component. The formalization used in the ASBLIntg is based on (Van den Berg et al., 2007).

$$(84) ASBLIntg = \{as1 \mid as1 \in AS \wedge \exists ac \in (ACI \cap ACBLDed): (ac, as1) \in Realization \wedge \exists as2 \in ASACIntg: (as2, as1) \in Aggregation\}$$

$$(83) ACBLDed = ACBL \backslash (ACBL \cap (ACPreL \cup ACIntg))$$

$$(85) ASACIntg = \{as | as \in AS \wedge \exists ac \in (ACIntgDed \cap W3C): (ac, as) \in Realization\}$$

$$(86) ACIntgDed = ACIntg \backslash (ACIntg \cap (ACPreL \cup ACBL))$$

The used ACBLDed set identifies the application components that only implement business logic. The ASACIntg identifies the application services provided by application components responsible for the integration with external components. These services are responsible for exposing the business logic application services. The integration components also support a determined protocol, which in this

case is the W3C protocol. The set ACIntgDed is used to define the application components dedicated only for the integration purpose.

An ASACIntg application example is illustrated in the Figure 48.

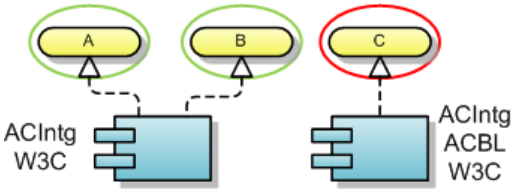

Figure 48 : ASACIntg application example.

In Figure 49 is also presented an ASBLIntg application example.

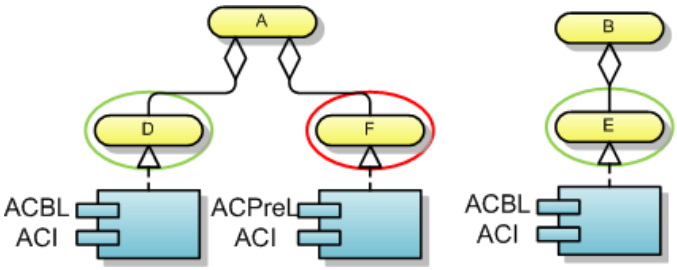

Figure 49 : ASBLIntg application example.

The other prescription to be addressed, prescribes that integrated external components only should implement presentation logic and requires a certain protocol for integration. In this proposal we consider, as previous, the W3C protocol.

Hereupon, the IESLD defines the exposed application services for integration that are used by external application components. These external components have to implement presentation logic and require the W3C protocol for integration. This set represents the compliant application services with the IESLD principle. The ACPreLDed set identifies the application components that only implement presentation logic. The ACU fuction used in IESLD, defines the application components that use a certain application service.

$$(87) IESLD = \{as | as \in ASBLIntg \wedge \big(ACU(as) \subseteq (ACE \cap W3C \cap ACPreLDed)\big)\}$$

$$(82) ACPreLDed = ACPreL \backslash (ACPreL \cap (ACBL \cup ACIntg))$$

$$(88) ACU(as) = \{ac | ac \in AC \wedge (as, ac) \in Used\ by\}$$

In Figure 50 is exemplified the application of IESLD set. It's also important to notice that the referred example is in line with the examples presented before.

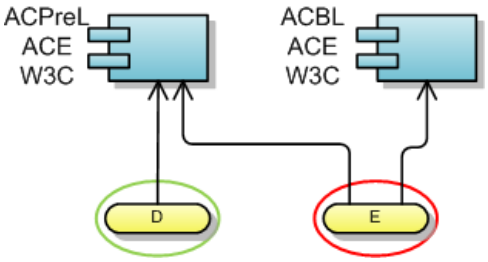

Figure 50: IESLD application example.

Chapter 4

Case Study

In this chapter, is explained how the proposal could be applied in a real case study. The presented case study consists in the representation of some specific architectures, which subsequently are used to evaluate their compliance with their guiding principles. For each architecture, through the respective architecture principle compliance analysis is expected the identification of the elements that address the principle intentions. For each analysis, besides the obtaining of the architecture compliant elements, it's expected to obtain the knowledge about the elements that need to be fixed to be considered compliant in the future. The architectures used represent partitions of Fidelidade EA. This case study corresponds to the demonstration step in DSRM (Section 1.3).

Fidelidade is a holding company that incorporates several insurance companies. The Fidelidade historic evolution is characterized by several merges with other companies which result in a large and complex EA. This complexity presents itself as a valuable point to apply the proposed compliance analysis, due to the possibility to find inconsistencies.

As referred in the principles related work, to evaluate the principles compliance we have to know what scope is impacted by them. So, it's expected that a determined architecture could be affected by a certain principle and other architecture by another.

Hereupon, initially the analyzed architectures are introduced and their respective EADs are also presented. Then, for each architecture the respective compliance analysis is applied and the resulting compliant elements are identified in green in the EADs. It's important to notice that the EADs used to describe a certain architecture are based on the principle perspectives that is being analyzed.

For each analysis the same obtained result is also presented by performing the same analysis automatically through the EAMS parameterization. This automatic analysis corresponds to the third step presented in the used approach (Chapter 3).

4.1 LEVE

The LEVE[5] is a solution for supplementary retirement savings. The LEVE solution is based on a retirement savings plan (PPR), which offers two investment options with different risk levels, where the client can invest in one of the options or allocate their investment between them. This solution allows managing all operations concerning the management of a PPR. These operations could be related with a simulation, subscription or change in a PPR and could also endorse the operations underlying the client information management.

[5] www.leve.pt

This solution is used to apply some of the proposed principle compliance analysis. This analysis pretends to evaluate, if LEVE addresses the principles prescriptions that guide its design. The analysis for each principle is performed independently, and for each one is presented a LEVE partition with the identification of the compliant elements resulting from the analysis application. It's also important to notice, that not all elements and relationships endorsed by LEVE are used in this application to avoid redundant analysis.

4.1.1 A.12 Principle Analysis Application

In Figure 51 is presented a view, where is described a LEVE partition. The described architecture should address all the A.12 principle intentions and for that reason is used to apply this analysis. So, to perform this compliance analysis the CSSCI set (Section 3.4.2) is applied to the referred view.

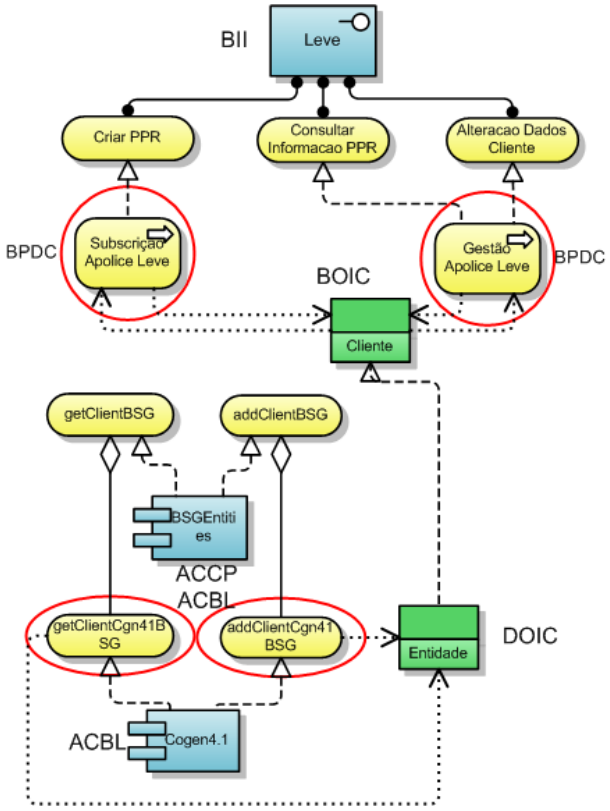

Figure 51: A.12 analysis demonstration in LEVE.

As can be seen, none of the elements that should be recognized as compliant are recognized since the component responsible for channel processing, is not only dedicated to that purpose and also implements business logic. The referred component is named by BSG Entities and corresponds to the middleware component that interfaces with business logic application components.

In this case, it's prescribed by the principle that BSG Entities only should implement channel processing logic. This non-compliance is identified when the ACSP set is applied, which doesn't recognize the BSGEntities as compliant. This non-recognition impacts the results of other sets that evaluate other prescriptions, specially the sets that endorse the communication between processes prescription (ASTCP, BOICDOT and BPDCDepC sets).

The presentation of this analysis, obtained in an automatic way, is not presented since the EAMS was not extended to address this compliance analysis.

4.1.2 A.14 Principle Analysis Application

In Figure 52 is presented a view, where is described a LEVE partition. The described architecture should address all the A.14 principle intentions and for that reason is used to apply this analysis. So, to perform this compliance analysis the DPS set (Section 3.5.2) is applied to the referred view.

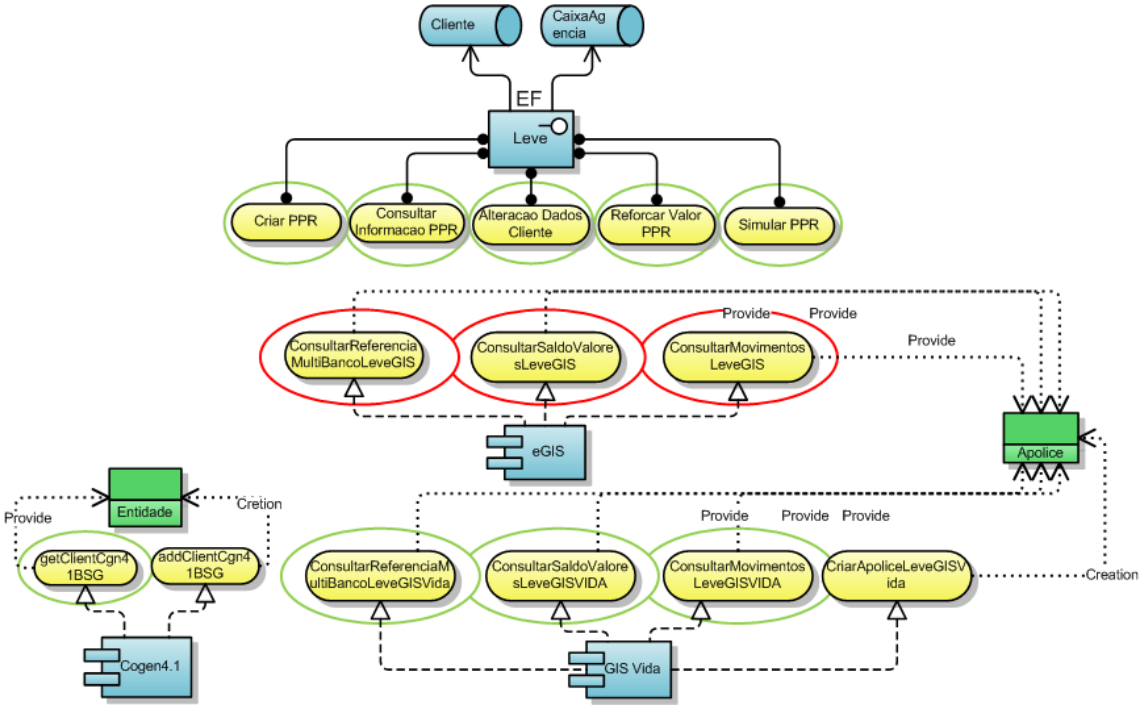

Figure 52: A.14 analysis demonstration in LEVE.

Through the principle analysis demonstrated above, it's perceptible that all application services realized by eGIS are non-compliant, since they provide data objects that are not created by eGIS. The non-recognition of these application services as compliant is performed when the ASDOASource set is applied. In this case, only the services realized by GIS Vida should provide that information, which represents the reason for their identification as compliant.

In Figure 53 is presented the same result, but is obtained through the automatic application of this analysis.

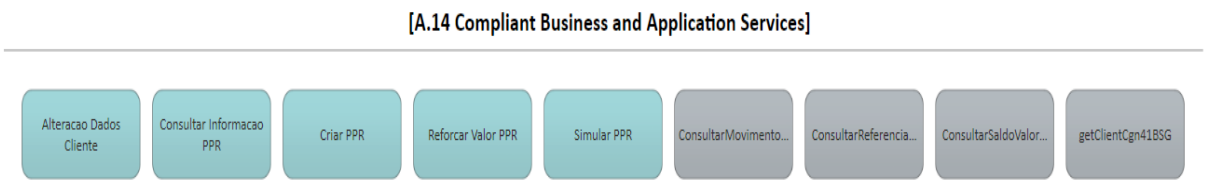

Figure 53: A.14 analysis automatic demonstration in LEVE.

4.1.3 A.28 Principle Analysis Application

In Figure 54 is presented a view, where is described a LEVE partition. The described architecture should address all the A.28 principle intentions and for that reason is used to apply this analysis. So, to perform this compliance analysis the AAM set (Section 3.7.2) is applied to the referred view.

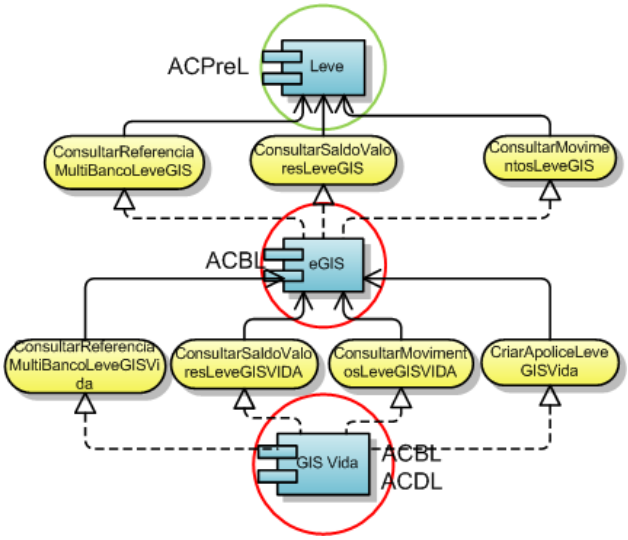

Figure 54: A.28 analysis demonstration in LEVE.

As can be seen, only Leve application component is identified as compliant. The reasons for the non-compliance are the follows.

- Unlike to what is intended by the AAM principle, the GIS Vida component is a monolithical application component that implements two different types of logic, which turn this element non-compliant. The non-recognition of this component as compliant is achieved through the MoAC1Log set application.
- The GIS Vida as previously referred, is a monolithical application component that implements two types of logic. The eGIS component uses services provided by GIS Vida and for that reason, it is not identified as a compliant element. This usage is against the layered structure intended by the principle, which results in the eGIS non-identification when the ACMUsage is applied, more precisely in the ACBLMUsage set.

In Figure 55 is presented the same result, but is obtained through the automatic application of this analysis.

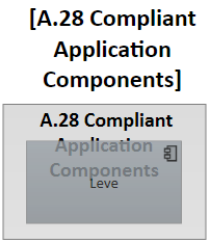

Figure 55: A.28 analysis automatic demonstration in LEVE.

4.1.4 A.42 Principle Analysis Application

In Figure 56 are presented two views, where is described a LEVE partition. The described architecture should address all the A.42 principle intentions and for that reason is used to apply this analysis. So, to perform this compliance analysis the PPBLS set (Section 3.9.2) is applied to the referred views.

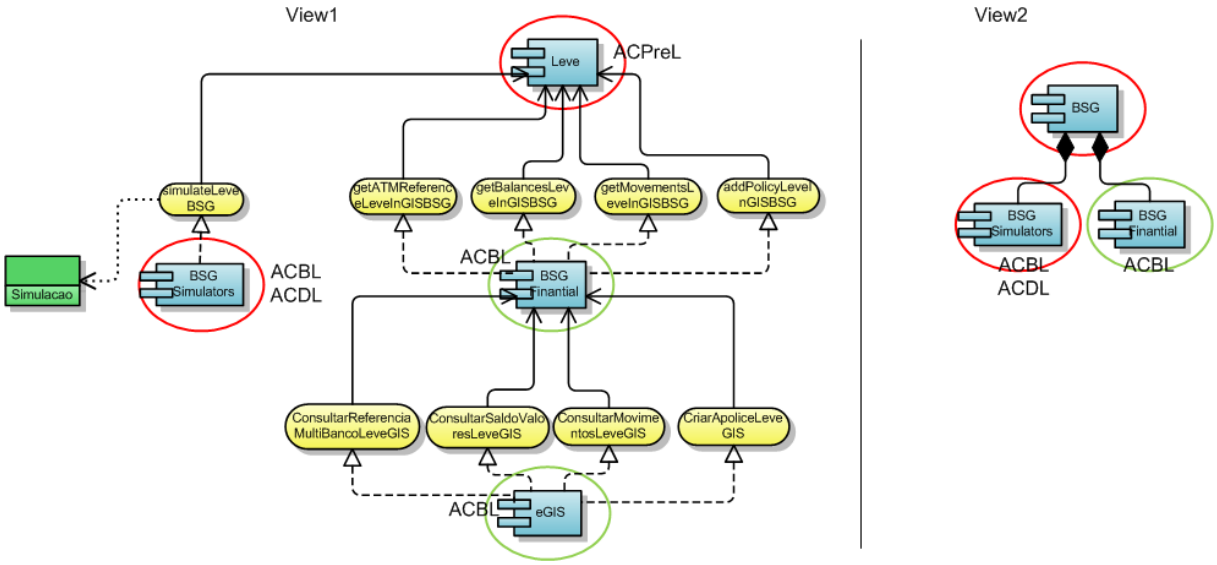

Figure 56 : A.42 analysis demonstration in LEVE.

The reasons for the non-recognition of BSG Simulators, Leve and BSG as compliant are:

- The BSG Simulators implements two different types of logic. The principle prescribes that monilithical applications only could implement one type of logic, which is not addressed by this component. The BSG Simulators non-recognition as compliant is performed by the MoAC1Log set application.
- The BSG Simulators, as previously referred is a monolithical application component that implements two types of logic. The Leve component uses services provided by BSG Simulators and for that reason, it is not identified as compliant since it's against the layered structure intended by the principle. This non-identification occurs when the ACMU set is applied, more precisely in the ACBLMU set.
- The BSG application component is not recognized as compliant, since it possesses in its composition the BSG Simulators component that doesn't comply with the principle intentions. The presence of BSG Simulators in the BSG composition is against the layered structure intended by the principle, which results in the BSG non-recognition when ACCompAC1L is applied.

In Figure 57 is presented the same result, but is obtained through the automatic application of this analysis.

Figure 57 : A.42 analysis automatic demonstration in LEVE.

4.2 CISN

CISN is the enterprise portal used in Fidelidade, which is considered the central point to access to the existent application services. The scope considered in this section, comprises application services provided through the referred enterprise portal and how they are exposed. It's also important to notice that not all elements and relationships that correspond to the CISN scope will be used in this application to avoid redundant analysis.

4.2.1 A.29 Principle Analysis Application

In Figure 58 is presented a view, where is described a partition of the services provided by CISN. The described architecture should address all the A.29 principle intentions and for that reason is used to apply this analysis. So, to perform this compliance analysis the AFTEP set (Section 3.8.2) is applied to the referred view.

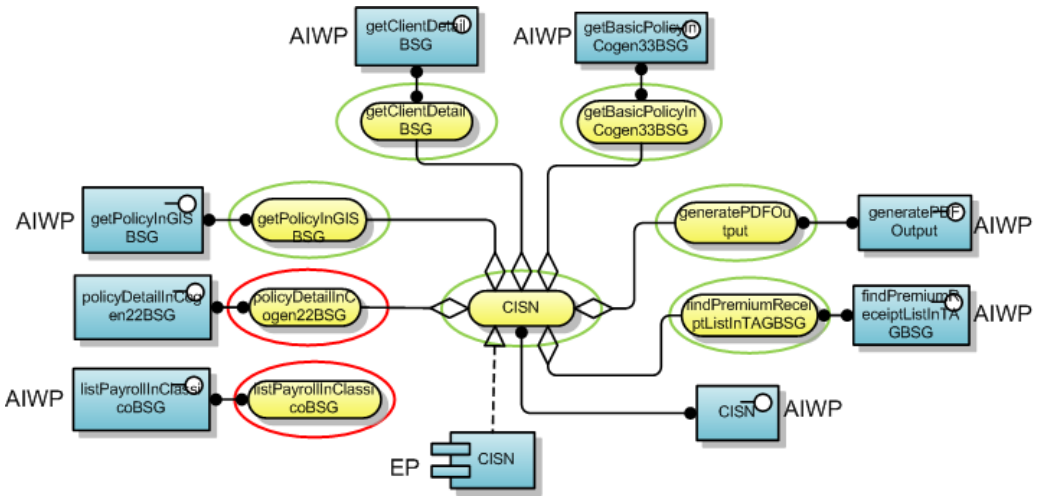

Figure 58: A.29 analysis demonstration in CISN.

In Figure 59 is presented the same result, but is obtained through the automatic application of this analysis.

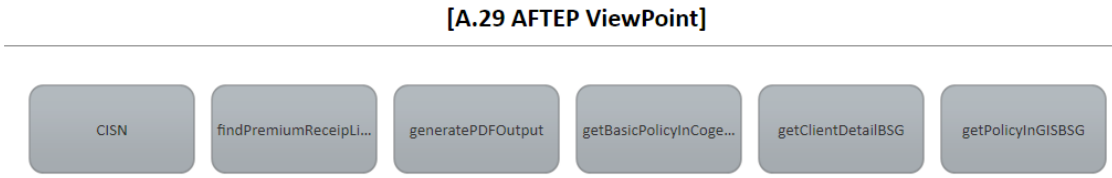

Figure 59: A.29 analysis automatic demonstration in CISN.

The reasons for the non-compliance verified are:

- The policyDetailInCogen22BSG service is not presented as a Web Part. This inconformity is recognized when the ASPWP set is applied, which results in the element non-presence in the referred set.
- The listPayrollInClassicoBSG service is not accessed through the enterprise portal, which is against the principle first implication. This non-conformance is considered with the ASTEP set application.

The non-presence of policyDetailInCogen22BSG and listPayrollInClassicoBSG in the ASPWP and ASTEP respectively, guarantees the non-recognition of these services as principle compliant elements.

4.3 WAI

The WAI is an application component in Fidelidade EA responsible for the replica synchronism. The scope endorsed by this section consists in the replica management, where are considered all the operations that impact a certain replica. It's also important to notice that in this scope will be considered a partition of the LEVE to enable a broader compliance analysis.

4.3.1 A.15 Principle Analysis Application

In Figure 60 is presented a view, where is described the scope previously mentioned. The described architecture should address all the A.15 principle intentions and for that reason is used to apply this analysis. So, to perform this compliance analysis the DMSA set (Section 3.6.2) is applied to the referred view.

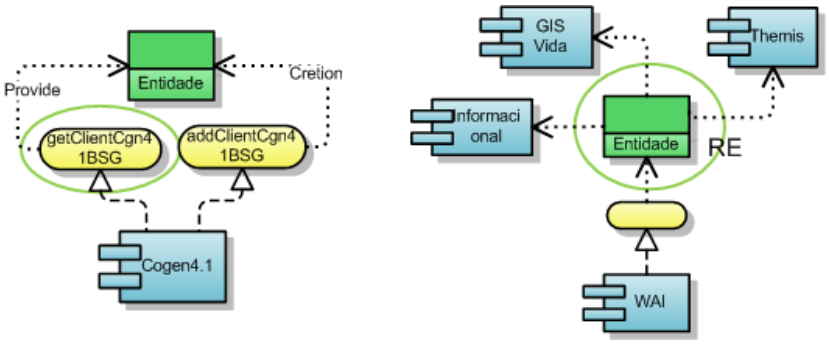

Figure 60: A.15 analysis demonstration.

As can be seen, the elements represented in the previous EAD, address all the prescriptions imposed by the DMSA principle and for that reason the getClientCgn41BSG and Entidade are recognized by the DMSA set.

The presentation of this analysis, obtained in an automatic way, is not presented since the EAMS was not extended to address this compliance analysis.

4.4 B2B Integration

The scope comprised in this section corresponds to the B2B integration between Fidelidade and one of their clients, where are considered the integrated services between them. These integrated services are used by the presented client to verify how the car insurance rates vary based on the geographical location and based on the car model. It's important to notice, that the name of the components relative to the Fidelidade client will be represented with a letter due to confidentiality reasons.

4.4.1 A.56 Principle Analysis Application

In Figure 61 is presented a view, where is described the B2B integration previously referred. The described architecture should address all the A.56 principle intentions and for that reason is used to apply this analysis. So, to perform this compliance analysis the IESLD set (Section 3.11.2) is applied to the referred view.

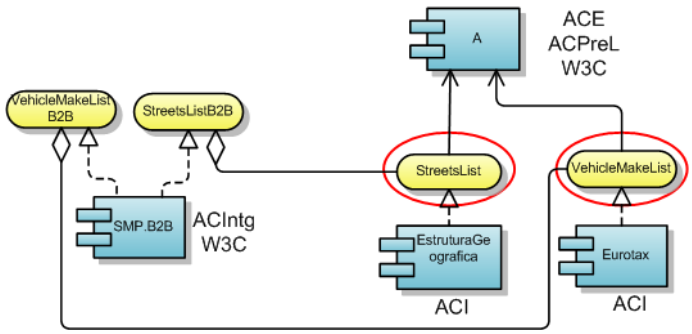

Figure 61: A.56 analysis demonstration.

In this case, the IESLD analysis results in an empty set. The integrated services (StreetList and VehicleMakeList) are not recognized as compliant since they are not realized by business logic application components, which is against the principle. This non-compliance is detected through the ASBLIntg set, which doesn't recognize the referred components and consequently results in an IESLD set without compliant services.

In Figure 62 is presented the same result, but is obtained through the automatic application of this analysis.

Figure 62 : A.56 analysis automatic demonstration.

4.5 Service Directory

The scope endorsed in this section contemplates several application services and how they are exposed. It also addresses the service directory, where application and infrastructure services are published.

4.5.1 A.43 Principle Analysis Application

In Figure 63 is presented a view, where is described the scope previously referred. The described architecture should address all the A.43 principle intentions and for that reason is used to apply this analysis. So, to perform this compliance analysis the ITSCTS set (Section 3.10.2) is applied to the referred view.

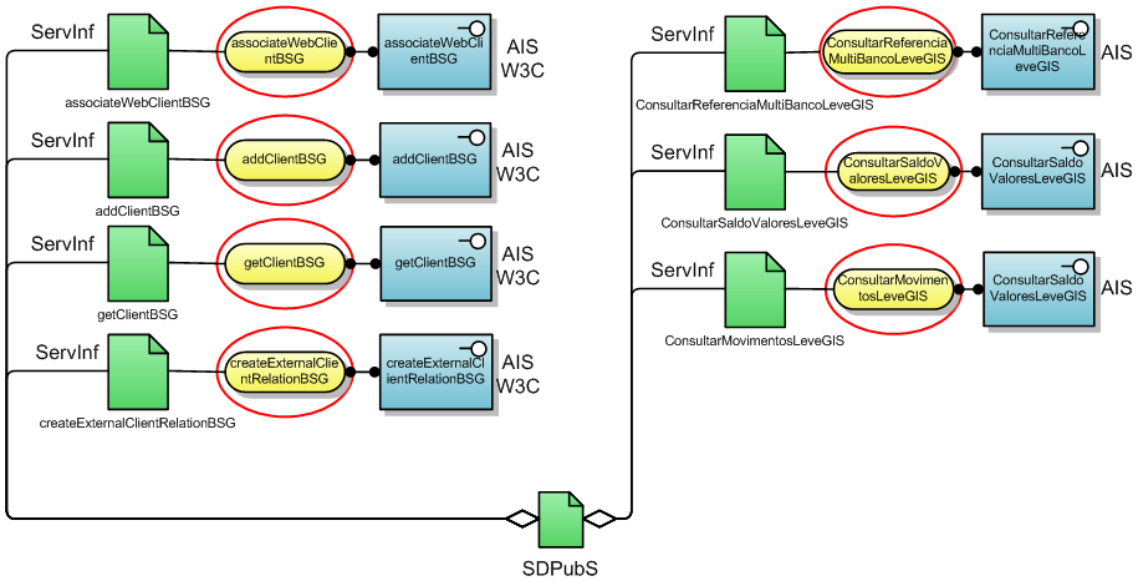

Figure 63 : A.43 analysis demonstration.

In this case, the ITSCTS analysis results in an empty set. The illustrated application services are not recognized as compliant, since the artifact where they are published is not managed by a system software (service directory) dedicated to manage that information.

As can be seen above, the Fidelidade EA doesn't possess a service directory (system software) to manage the artifact responsible to aggregate the information about the published services, which is against the principle intention. The service directory non-recognition is achieved through the SDInfPubS set.

The ConsultarReferenciaMultiBanco, ConsultarSaldoValoresLeveGIS and ConsultarMovimentosLeveGIS services are also not recognized as compliant for another reason. The ITSCTS principle prescribes that application services should be provided through services formally defined, adhering to standard protocols. This prescription is not achieved by the referred services, which results in their non-identification when the ASServP set is applied.

The presentation of this analysis, obtained in an automatic way, is not presented since the EAMS was not extended to address this compliance analysis.

4.6 Front and Back Office

The scope endorsed by this section, addresses several business processes and the application components that support them. These selected elements should address the A.56 principle intentions and for that reason, this scope is used to apply this compliance analysis.

4.6.1 A.11 Principle Analysis Application

In Figure 64 is presented a view, where is described the scope previously referred. To perform the A.11 compliance analysis the FOSBO set (Section 3.3.2) is applied to the referred view.

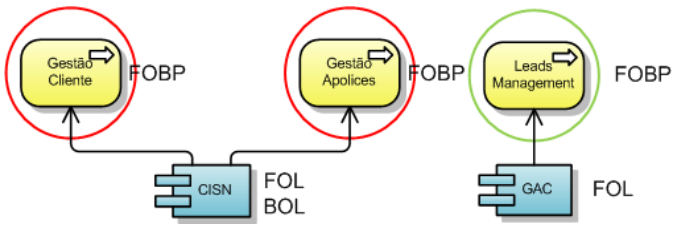

Figure 64 : A.11 analysis demonstration.

Through the FOSBO set, the Lead Management is the only business process identified as compliant. The Gestão Cliente and Gestão Apolices represent front-office business processes and according to the principle, they should be supported by application components that only implement front-office logic. However, this prescription is not achieved since CISN is not only dedicated to front-office.

The CISN non-recognition as a front-office logic dedicated component is performed when ACFOL set is applied. This non-recognition results in the non-identification of Gestão Cliente and Gestão Apolices processes as principle compliant elements.

In Figure 65 is presented the same result, but is obtained through the automatic application of this analysis.

Figure 65 : A.11 analysis automatic demonstration.

Chapter 5

Evaluation

This chapter corresponds to the evaluation step in the DSRM (Section 1.3). It is here where the achieved results from the proposal demonstration are compared with the contributions underlying this research.

This evaluation method will consider the following steps:

- **Results Discussion.** This step will present considerations about the obtained results.
- **Results Analysis.** In this section, will be analyzed if the research questions (Section 1.1) are or not positively answered and if the research goal and contributions (Section 1.2) are or not achieved.

5.1 Results Discussion

In Section 4, the compliance analysis for nine principles was performed to demonstrate the feasibility of our proposal. The used EADs are provided by Fidelidade and represent partitions of specific architectures that should address the respective principles. The EADs used are simple, but demonstrate the applicability of our proposal. More complex EADs could be used, but several redundant evaluations would be performed.

Through the compliance analysis previously applied, some general considerations about the obtained results could be made:

- It's possible to determine the principle expected impact in the EA. In the research proposal, this impact represents the basis for the principle compliant elements identification. The principle expected impact, initially endorses the principle perspective definition and then, consists in the principle formalization, where are analyzed the architecture structures that characterize each principle.
- If a certain EAD or EADs possess the elements and relationships needed to perform the compliance analysis of a certain principle, it's possible to recognize its compliant elements. Hereupon, the principle compliance evaluation could be performed through the use of EADs.
- The proposed analysis could be performed automatically through the use of an EAM tool. The principle formalization underlying the proposed compliance analysis, is defined in a way that enables its parameterization in an EAM tool.
- The compliance analysis for each principle could be applied to non-ArchiMate based EADs, since the underlying metamodel endorses the elements and relationships used in the respective compliance analysis.

The recognition of the principle compliant elements through the proposal application, also enables to acquire the knowledge of what the non-compliant elements are. Below, it's performed the match between some specific unconformities and the Fidelidade EA evolution, to make explicit the justification for the obtained results. These reasons are the follows.

- Several application components were developed before the emergence of application modularity principles, where logic separation is advised for an easier application maintenance and reuse. This past legacy represents the reason for the non-recognition as compliant of some elements in the application of A.12, A.28 and A.42 compliance analysis.
- Concerning the A.14 analysis (Figure 52), it was identified a redundancy in Fidelidade middleware layer. The existence of this redundancy is known by the architects and is explained by the acquisition of eGis and GIS Vida components as a package. When other components want to obtain the Apolice object from services provided by GIS Vida they can't, since eGIS is responsible for the communication with GIS Vida. So, the Apolice object only could be provided through eGIS, which does not represent its application source. Consequently, the services realized by eGIS are not recognized as compliant.
- Regarding the A.43 analysis, it was identified the lack of a dedicated software systems to manage the service directory information. This lack was previously identified by Fidelidade architects and for that reason, it is being developed a dedicated system software to surpass the identified non-conformity.
- The separation between front and back office represents a growing concern by Fidelidade. However, in some cases is difficult to achieve the referred separation, especially at the application component level. This difficult emerges from the past legacy, where the separation of front and back office logics in the application development was not intended. This fact represents the reason for the unconformities recognized in A.11 compliance analysis.

Hereupon, the compliance analysis proposed by this work presents itself as a mechanism to evaluate the EA, based on architecture principles. The scope used to apply this analysis could endorse solution architectures, as-is or to-be of the EA.

5.2 Results Analysis

In this section, the obtained results are analyzed to verify if the questions underlying this research are positively or negatively answered and also, if the proposed contributions are accomplished.

The question **Q1** is answered through the related work study (Section 2.2). This study endorses the architecture principles field, where is recognized their importance and more specifically, it's identified a gap in the existing techniques to verify the EA principle compliance. This gap represents the lack of techniques to verify the principle compliance through EADs. In the literature review of the existing architecture analyses to evaluate the EA quality, it is also perceptible the non-consideration of architecture principles to evaluate the EA.

Regarding the question **Q2**, its answer is positive, which could be verified by the results obtained in the Fidelidade case study (Section 4). However, it's important to notice that to perform a determined principle compliance analysis, all the elements and relationships endorsed by the respective analysis have to be considered in the used EADs.

The questions **Q3** and **Q4** are intimately connected with the **Q2** question. The answers of these two questions have a good share on the answer of the **Q2** question.

The question **Q3** is answered by the perspectives definition regarding each compliance analysis. These perspectives identify the elements and relationships impacted by the principle and represent what is needed to perform the respective analysis. The Table 14 (Appendix) is also endorsed in this answer.

The question **Q4** is answered, when are defined the characteristics that should be obeyed by the elements and relationships impacted by the principle. These characteristics are obtained through the principle prescriptions, which are formalized in the principle expected impact. This formalization enables to identify the elements and relationships aligned with the principle intentions.

Hereupon, the contributions proposed by this research are achieved, which positions this research as a contribution in the principle compliance management.

Chapter 6

Conclusion

Over time the principle role has gained a greater relevance in the proper EA evolution. However, sometimes the principle application is influenced by contextual factors, which could comprise its effectiveness (Greefhorst and Proper, 2011). In this case, a deviation from the EA expected impact may emerge, which could be against the principle purpose.

The architecture principles present themselves as the rationale for several architecture elements and relationships in the EA models (Greefhorst and Proper, 2011), which positions the EADs as useful artifacts to analyze the EA compliance with its guiding principles. However, due to architecture principles novelty an architecture analysis to evaluate the EA compliance using EADs still lacks. This lack represents the research problem underlying this investigation.

To surpass the identified problem, this work proposes an architecture analysis based on architecture principles. The proposed analysis enables to identify the compliant elements of an EAD with the EA guiding principles. The principle expected impact and the ArchiMate language provide the basis for the approach underlying the proposed analysis. Initially, the architecture perspectives are defined, which provide the elements and relationships impacted by the principle. Then, for each perspective are formally defined the conditions prescribed by the selected principle. This formalization is used to verify if the elements and relationships of a certain perspective respect the principle prescriptions and consequently, enable to identify the compliant elements.

The research problem motivation and the concepts underlying the proposed analysis are endorsed by the literature study, which addresses the ArchiMate, the architecture principles and the architecture analysis fields.

Then, the research proposal feasibility was demonstrated in real architectures, where compliant elements were identified. These specific architectures were provided by Fidelidade and should address at least one of the architecture principles considered by the research proposal. Hereupon, for each architecture were applied the respective compliance analysis.

Through the obtained results, the compliant elements were identified and the knowledge about the non-conformities was also gained. These non-conformities are then justified, based in Fidelidade EA past evolution to guarantee the correctness of the obtained results. Hereupon, the obtained results enable to surpass the identified problem and for that reason this research represents a contribution for the architecture principle field.

6.1 Communication

This section corresponds to the communication step of DSRM (Section 1.3), which consists in the research proposal and respective contributions communication to the adequate audiences. Regarding this research, the submission of scientific publications was the chosen way to achieve the referred audience.

During this research, it was submitted and subsequently accepted the following publication:

- Alves, J., Vasconcelos, A., Sousa, P.(2014). Architecture Principle Compliance Analysis. In: *16th International Conference on Enterprise Information Systems (ICEIS 2014)*.

This publication endorses all sections endorsed by this research. However, relatively to the proposed compliance analysis, only the A.14 and A.28 compliance analyses are considered.

6.2 Main Contributions

In this section are revised the contributions proposed by this research (Section 1.2). The research contributions are the follows.

- (Main) The proposal of an architecture analysis to evaluate the EA compliance with its guiding principles. This analysis is based on the recognition of architecture structures that represent the principle expected impact. This recognition is based on the formalization of these structures in ArchiMate, which enables to identify the compliant elements in the respective EADs.
- (Secondary) The proposal of a mechanism to perform the compliance analysis previously mentioned in an automatic way. This mechanism consists in the enterprise architecture management (EAM) tool parameterization to recognize the principle expected impact, and consequently recognize the principle compliant elements. This mechanism enables to apply the principle compliance analysis to a broader and complex scope.
- (Secondary) To demonstrate the compliance architecture analysis it's used a case study provided by Fidelidade. This case study besides being used to demonstrate the proposal feasibility is also valuable to the architecture principles field due to its novelty.
- (Secondary) To understand what could be addressed by this investigation, the literature study related with architecture principles and architecture analysis are presented. This study presents itself as a contribution, due to non-existence of literature that relates these two fields.

6.3 Reflections

In this section are highlighted some issues that an organization has to be aware if it pretends to apply the research proposal.

First of all, the organization must be aware of the principle relevance in the architecture evolution. If the principle purpose is not assimilated and shared by all stakeholders, the practices endorsed by the principle life-cycle could be ineffective. So, it's essential the integration of the architecture principle

concept in the organizational culture, in order to be recognized by all its importance. This represents the first step to consider, when the organization pretends to apply the principle related practices, and consequently, when pretends to apply our proposal.

Then, for each principle defined, the respective stakeholders have to perceive how their work is affected and what could be their impact in the architecture. The stakeholder's perception of the principle impact has to be aligned with the principle prescriptions. It is here that our proposal could be used to guarantee this essential alignment.

As previously mentioned, the research proposal is closely linked how the architecture could be described. As can be seen, the compliance analysis proposed by this research is dedicated to ArchiMate and for that reason, the ArchiMate should be considered by the organization to describe its architecture. Relatively to the organizations that consider dedicated metamodels to describe their architecture, if they want to perform this analysis, it's required that the underlying metamodel endorse the elements and relationships used to verify the respective compliance analysis. The referred elements and relationships must possess the same meaning of the ArchiMate elements used.

Finally, to apply the proposed analysis, it's also important for the organization reflect about the following points:

- The scope that will be evaluated. It's important to define the architecture scope that will be evaluated. This scope could endorse the compliance analysis of a solution architecture, until the compliance evaluation of the architecture "to-be".
- The role definition. It's important to define who will be responsible to apply the referred analysis.
- When the proposed analysis should be applied. It's important to perceive if the architecture compliance should be applied in the early stages of a project or not, for example.

If an organization addresses the assumptions referred above, the conditions are set to apply the research proposal with effective results.

6.4 Limitations

The architecture principles are characterized for being an immature field. Due to this fact, there are few organizations that already adopt the architecture principle concept to evolve their EA. Hereupon, the proposed solution only is demonstrated in one organization, which represents a limitation of our work.

The contextual factors influence in the principle interpretation could also represent a limitation. The interpretation made in the proposed analysis could differ in some points from other interpretations. Consequently, from different interpretations could result different principles formalization.

Through the proposed analysis, the architecture structures that represent the principle expected impact are analyzed. As previously seen, the analysis for each principle consists in a set that identifies

the principle compliant elements. This set is composed by subsets that also could be composed by other subsets. These subsets analyze subparts of the architecture structures that characterize the principle. Through these subsets, compliant elements are also identified. However, these elements not always appear in the set that defines the principle compliant elements. The set that defines the principle compliant elements, only identify certain types of EA elements, which could not endorse the elements identified by the subsets previously mentioned. This fact could be considered a limitation of our proposal.

6.5 Future Work

In this section, are highlighted future directions that could be endorsed to evolve this research and related areas. The presented directions are aligned with the limitations mentioned in Section 6.3 and have the purpose to overcome them.

As mentioned before, the architecture principles and architecture patterns present some similarities that could be explored. The architecture patterns, such as architecture principles, represent the rationale for certain elements and relationships in the EA. The architecture structures that characterize a pattern could or not respect the expected impact of a certain principle. In this case, the determination of what patterns are compliant with the principles analyzed, can be exploited. This direction would require the patterns formalization in ArchiMate, as realized in (Šaša and Krisper, 2011). This relationship between patterns and principles are also referred by Hoogervorst (2009), which states that patterns are specific standards which form a subset of principles. The same idea is shared by (Greefhorst and Proper, 2011), which refers that a principle could be realized by a certain pattern.

The research proposal application to other organizations could represent another future direction to address. This could result in the principle formalization extension, as consequence of other interpretations and other EADs. This extension could consist in a different way of formalizing the principle impact and therefore may result in a more complete analysis.

Bibliography

Aier, S., Fischer, C., Winter, R. (2011). Construction and Evaluation of a Meta-Model for Enterprise Architecture Design Principles. *In: The 10th international conference on Wirtschaftsinformatik (WI 2011)*, pp. 637-644.

Aitken, C. (2010). EA management patterns for future state design. *In: 2nd European workshop on patterns for enterprise architecture management (PEAM2010)*, pp. 267-280.

Chorus, G.J.N.M., Janse, Y.H.C., Nellen, C.J.P., Hoppenbrouwers, S.J.B.A., Proper, H.A. (2007). *Formalizing architecture principles using object-role modelling*. Available at: http://vianovaarchitectura.nl/page/formalizing-architecture-principles-using-object-role-modelling

Gama, N., Mira da Silva, M., Caetano, A., Tribolet, J.M.N.S. (2007). Integrar a arquitectura organizacional na arquitectura empresarial. *In: 7ª Conferência da Associação Portuguesa de Sistemas de Informação.*

Greefhorst, D., Proper, E. (2011). *Architecture Principles: The Cornerstones of Enterprise Architecture*. Berlin: Springer.

Van den Berg, H., Bosma, H., Dijk, G., Van Drunen, H., Van Gijsen, J., Langeveld, F., Luijpers,J., Nguyen, T., Oosting, G., Slagter, R., Willemsz, E. (2007). *ArchiMate Made Practical.* Available at: http://www.archimate.nl/en/start_using_archimate/good_practices.html

Hevner, A., March, S., Park, J., & Ram, S. (2004). Design Science in Information Systems Research. *Management Information Systems Quarterly* 28(1), pp. 75 – 106.

Hoogervorst, J.A.P. (2004). Enterprise architecture: Enabling integration, agility and change. *International Journal of Cooperative Information Systems* 13(3), pp. 213-233.

Hoogervorst, J.A.P. (2009). *Enterprise governance and enterprise engineering*. Diemen: Springer.

Johnson, P., Lagerström, R., Närman, P., Simonsson, M. (2007). Enterprise architecture analysis with extended influence diagrams. *Information Systems Frontiers* 9, pp. 163.180.

Lankhorst, M.M. (2004). Enterprise architecture modelling-the issue of integration. *Advanced Engineering Informatics* 18, pp. 205-216.

Lankhorst, M. (2009). *Enterprise Architecture at Work: Modelling Communication and Analysis*. 2nd ed. Berlin: Springer.

Lindström, A. (2006). On the syntax and semantics of architectural principles. *In: Proceedings of the 39th Hawaii international conference on system sciences*, pp. 178b-178b.

OMG (2009). *Business Process Model and Notation (BPMN)*. Available at: http://www.omg.org/cgi-bin/doc?dtc/09-08-14

Op't Land, M., Proper, H.A., Waage, M., Cloo, J., Steghuis, C. (2009). *Enterprise architecture—creating value by informed governance*. Berlin: Springer.

Peffers, K., Tuunanen, T., Rothenberger, M. A., & Chatterjee, S. (2008). A Design Science Research Methodology for Information Systems Research. *Journal of Management Information Systems* 24(3), pp. 45-78.

Richardson, G.L., Jackson, B.M., Dickson, G.W. (1990). A Principles-Based Enterprise Architecture: Lessons from Texaco and Star Enterprise. *Management Information Systems Quarterly* 14(4), pp. 385-403.

Šaša, A., Krisper, M. (2011). Enterprise architecture patterns for business process support analysis. *Journal of Systems and Software* 84 ,pp.1480-1506.

Staron, M., Kuzniarz, L., Wohlin, C. (2006). Empirical assessment of using stereotypes to improve comprehension of UML models: a set of experiments. *Journal of Systems and Software* 79, pp. 727-742.

The Open Group (2012). *ArchiMate 2.0 Specification*.

The Open Group (2009). *TOGAF Version 9.1*. Available at: http://pubs.opengroup.org/architecture/togaf9-doc/arch/

Van Bommel, P., Hoppenbrouwers, S.J.B.A., Proper, E. H.A., Van der Weide, T.P. (2006). Giving Meaning to Enterprise Architectures - Architecture Principles with ORM and ORC. *Meersman, R., Tari, Z., Herrero, P. et al. (eds.): OTM Workshops 2006* 4278, pp. 1138-1147.

Van Bommel, P., Buitenhuis, P.G., Hoppenbrouwers, S.J.B.A., Proper, H.A. (2007). Architecture principles—a regulative perspective on enterprise architecture. *Reichert M, Strecker S, Turowski K (eds) Enterprise modelling and information systems architectures (EMISA2007)* 119, pp.47-60.

Vasconcelos, A. (2007). *Arquitecturas dos Sistemas de Informação: Representação e Avaliação.* PhD thesis. Instituto Superior Técnico, Universidade Técnica de Lisboa.

Vieira, A., Costa, L., Amaro, P., Amorim, L., Nunes, P., Pina, M., Miguel, L., Pereira, C., Sousa, P. (2004). Arquitectura empresarial e sistemas de gestão e qualidade. In: *Proceedings of the QUATIC'2004 Quality: the bridge to the future in ICT*.

Vieira, T. (2012). *Evaluating Enterprise Architectures: From Principles to Metrics.* Msc thesis. Instituto Superior Técnico, Universidade Técnica de Lisboa.

Appendix

Symbols

BP	Set of all business process in the EAD	$(a, b) \in Realization$	a is related to b with the Realization relationship: a realizes b
AS	Set of all application services in the EAD	$(a, b) \in Usedby$	a is related b with the Used By relationship: a is used by b
AC	Set of all application components in the EAD	$(a, b) \in Provide$	a is related to b with the Provide relationship: a provides b
BR	Set of all business roles in the EAD	$(a, b) \in Creation$	a is related to b with the Creation relationship: a creates b
BI	Set of all business interfaces in the EAD	RE	Set of all replicas in the EAD: $RE \subseteq DO$
DO	Set of all data objects in the EAD	SM	Set of all application components responsible for replica synchronism in the EAD: $SM \subseteq AC$
FOBP	Set of all front-office business processes in the EAD: $FOBP \subseteq BP$	ACPreL	Set of all application components that implement presentation logic of an EAD: $ACPreL \subseteq AC$
BOBP	Set of all back-office business processes in the EAD: $BOBP \subseteq BP$	ACProL	Set of all application components that implement process logic of an EAD: $ACProL \subseteq AC$
ART	Set of all artifact of an EAD	ACBL	Set of all application components that implement business logic of an EAD: $ACBL \subseteq AC$
BO	Set of all business objects of an EAD	ACDL	Set of all application components that implement presentation logic of an EAD: $ACDL \subseteq AC$
FOL	Set of all application components that implement front-office logic in the EAD: $FOL \subseteq AC$	AI	Set of all application interfaces in the EAD
BOL	Set of all application components that implement back-office logic in the EAD: $BOL \subseteq AC$	EP	Set of all enterprise portals in the EAD: $EP \subseteq AC$
CR	Set of all costumers of an EAD: $CR \subseteq BR$	WP	Set of all web parts in the EAD: $WP \subseteq AI$
EF	Set of all electronic forms of an EAD:	BPIC	Set of all channel independent

	$EF \subseteq BI$		business processes of an EAD: $BPIC \subseteq BP$
BPDC	Set of all channel dependent business processes of an EAD: $BPDC \subseteq BP$	BII	Set of all Internet business interfaces of an EAD: $BII \subseteq BI$
ACCP	Set of all channel processors of an EAD: $CP \subseteq AC$	BIT	Set of all Telephone business interfaces of an EAD: $BIT \subseteq BI$
BOIC	Set of all channel independent business objects of an EAD: $BOIC \subseteq BO$	DOIC	Set of all channel independent data objects of an EAD: $DOIC \subseteq DO$
BODC	Set of all business objects channel dependent of an EAD: $BODC \subseteq BO$	DODC	Set of all channel dependent data objects of an EAD: $DODC \subseteq DO$
BS	Set of all business services of an EAD.	AIS	Set of all application interfaces that correspond to a service of an EAD: $AIS \subseteq AI$
IS	Set of all infrastructure services of an EAD.	IIS	Set of all infrastructure interfaces that correspond to a service of an EAD: $IIS \subseteq II$
PubS	Set of all infrastructures services that enable to publish a service of an EAD: $PubS \subseteq IS$	II	Set of all infrastructure interfaces of an EAD
SeaS	Set of all infrastructures services that enable to search a service of an EAD: $SeaS \subseteq IS$	SD	Set of all system software that represent a service directory of an EAD: $SD \subseteq SS$
SDPubS	Set of all artifacts that correspond to information about published services of an EAD: $SDPubS \subseteq ART$	SS	Set of all system software of and EAD
ServInf	Set of all artifacts that correspond to the information publish about a service of an EAD: $ServInf \subseteq ART$	ACI	Set of all internal application components of an EAD: $ACI \subseteq AC$
W3C	• Set of all application interfaces defined with W3C protocol of an EAD: $W3C \subseteq AI$ • Set of all infrastructure interfaces defined with W3C protocol of an EAD: $W3C \subseteq II$ • Set of all application components that require the use of services defined in W3C of an EAD: $W3C \subseteq AC$	ACE	Set of all external application components of an EAD: $ACI \subseteq AC$

	• Set of all application components that integrate applications components based on W3C of an EAD: $W3C \subseteq AC$		
$(\boldsymbol{a}, \boldsymbol{b})$ $\in \boldsymbol{Access}$	a is related to b with the Access relationship: a accesses b	ACIntg	Set of all application components responsible for integration between organizations of an EAD: $ACIntg \subseteq AC$
$(\boldsymbol{a}, \boldsymbol{b})$ $\in \boldsymbol{Aggregation}$	a is related to b with the Aggregation relationship: a aggregates b	(a, b) $\in Composition$	a is related to b with the Composition relationship: a is composed of b
$(\boldsymbol{a}, \boldsymbol{b})$ $\in \boldsymbol{Association}$	a is related to b with the Association relationship: a is associated with b	(a, b) $\in Assignment$	a is related to b with the Assignment relationship: a is assigned to b

Table 14: Symbols for principle formalization.