Information System Architecture Evaluation: From Software to Enterprise Level Approaches

André Vasconcelos, Pedro Sousa, José Tribolet

CEO - Centro de Engenharia Organizacional,
INESC – Instituto de Engenharia de Sistemas e Computadores
and
Department of Information Systems and Computer Engineering, Instituto Superior Técnico,
Technical University of Lisbon, Portugal

E-mails: { andre.vasconcelos, pedro.sousa, jose.tribolet }@ceo.inesc.pt

Abstract

In order to ensure that technology supports business needs and that IT investments deliver the desired value, it is fundamental to define an Information System Architecture (ISA) and measure its accurateness to the business model and existing technologies. Thus, in this paper we are concern on evaluating ISA by measuring its qualities (relevant at enterprise level).

Since software architecture (SA) is part of the information system architecture and the evaluation topic is a quite mature issue on the software engineering domain, we enumerate and classify several software evaluation approaches in order to consider its applicability to ISA evaluation. Therefore, in this paper, we present and classify the most significant software evaluation issues, namely: software qualities, software evaluation approaches, and software metrics.

Our preliminary findings indicate that: the quality attributes relevant for SA evaluation are generally applicable for ISA evaluation, the SA evaluation approaches are also useful for ISA evaluation, and the SA metrics are not applicable to ISA evaluation.

In this paper we propose a set of metrics for ISA evaluation, considering the most experienced and tested software engineering metrics. We apply the ISA evaluation approach, qualities and metrics to a health-care project case study.

Key-words: Evaluation, Metrics, Information System Architecture (ISA), Enterprise Architecture, Software Architecture (SA), CEO Framework, Health-care ISA

1. Introduction

Nowadays organizations business environment presents new challenges where information, innovation and the agility to continuously rethink business are key success factors (Nilsson et al. 1998). In order to address these new business needs, organizations usually try to find on IT the solution. Despite the significant technological progresses, organizations investments on IT frequently do not provide the expected returns (Boar 1999).

In this paper we argue that, in order to ensure that technology supports business needs and that IT investments deliver the desired value, it is fundamental to define an Information System Architecture (ISA) and measure its accurateness to the business model and existing technologies. Thus, in this paper we are concern on evaluating ISA by measuring its qualities.

The authors believe that ISA – a distinct concept from Software Architecture (SA) – has a vital role in the development of Enterprise Information Systems that are capable of staying fully aligned with organization strategy and business needs.

However, having an ISA does not make clear to the architect: how much an IS (or IS components) will respect some business properties/qualities; or which qualities (and how) an architectural decision might affect; or which ISA decision is more adequate… (Vasconcelos et al. 2004c)

The ISA evaluation is concern on inferring the ISA accurateness to a business model and existing technologies – for example, the alignment between business and IT is an issue that should be considered when determining the ISA quality.

In order to address the ISA evaluation topic, considering the similar roots between Information System and Software Engineering areas and that the evaluation topic is a quite mature issue on the Software Engineering domain, in this paper we will enumerate and classify several software evaluation approaches and analyze its suitability for ISA evaluation.

The next section of this paper presents the major concepts relevant for ISA evaluation, namely the enterprise architecture definitions and some evaluation definitions. Section 3 describes Software Engineering approach for SA evaluation (as SA evaluation methodologies, software qualities and software metrics). In section 4 we propose our ISA Evaluation approach. Section 5 presents a case study where we apply the evaluation metrics to an ISA in the Portuguese health-care system. Finally, section 6 draws some conclusions and presents future work.

2. Key Concepts

In this section we introduce the main notions, definitions and problems on enterprise, business, information system and software architectures, which will support the remaining sections of this paper.

2.1. Enterprise Architectures

ISA is a part of a vaster field of architectures and models relevant for the organization. Considering the architectural scope and level, one can distinguish the following architectures:

- Enterprise Architecture
- Information System Architecture (ISA).
- Software Architecture (SA)

SA main study area is on how programs or application components are internally built (Carnegie 2005). At this level it is import to consider the objects and classes needed for implementing the software. SA is a quite stable and mature field.

Enterprise Architecture is a group of models defined for getting a coherent and comprehensible picture of the enterprise (Tissot et al. 1998). The models define different “perspectives or viewpoints from which the company is considered, focusing on some aspects and ignoring others in order to reduce complexity” (Vernadat 1996). Thus, a model of the company can contain several activity, process, organization, information and behaviour diagrams of the company.

Enterprise architecture is considered a vaster concept than ISA, which includes business strategies and processes, besides Information System (IS) models that support them. Usually, at enterprise

architecture level, IS are consider “simple” resources used in business (as people, equipment and material, etc.) – e.g., Eriksson et al. (2000) and Marshall (2000).

Finally, ISA addresses the representation of the IS components structure, its relationships, principles and directives (Garlan et al. 1995), with the main propose of supporting business (Maes et al. 2000).

Quoting IEEE Architecture Working Group (1998), ISA level should be high. Thus, ISA is distinguished from software representation and analysis methods (as E-R diagrams, DFD), presenting an abstraction of internal system details and supporting organization business processes (Zijden et al. 2000).

ISA usually distinguish three aspects, defining three “sub architectures” (Spewak et al. 1992):

- Informational Architecture, or Data Architecture, represents the main data types that support business (Spewak et al. 1992), (DeBoever 1997).
- Application Architecture, defines applications needed for data management and business support.
- Technological Architecture, represents the main technologies used in application implementation and the infrastructures that provide an environment for IS deployment – as network, communication, distributed computation, etc. (Spewak et al. 1992), (Open 2003)

ISA description is a key step in ensuring that IT provides access to data when, where and how is required at business level (Spewak et. al. 1992).

However, having an ISA does not ensure these benefits just by existing; the representation of the information systems and its dependencies to business is a necessary, but not sufficient, step towards addressing key problems as the IS integrity and flexibility, IS ROI, IS and business alignment, among others.

2.2. Evaluation

Clements et al (2002) ask “How can you be sure whether the architecture chosen for your software is the right one? How can you be sure that it won’t lead to calamity but instead will pave the way through a smooth development and successful product?”. The architecture is the foundation for deducing the system quality attributes (as modifiability, performance, availability, reliability). The process of analyzing and deducting the architectural potential for implementing a system capable of supporting the major business requirements and identifying major risks and trade-offs is evaluation main concern.

The evaluation process will consider the architectural attributes, the properties that characterize the system – e.g., CPU speed (in a technological architecture), or the development language (in a SA).

The characteristics that we pretend to verify in the architecture are defined as quality attributes (or quality requirements or external attributes) – such as modifiability, performance, availability, reliability.

In the evaluation process the quantitatively interpretation of the observable architectural attributes are defined as metrics – e.g., number of lines of code, function points.

3. Software Architecture Evaluation

In this section we present the software engineering approach to software evaluation. Some of the SA evaluation methods introduced in this section are the foundations for the ISA evaluation approach proposed in subsequent sections of the paper.

3.1. Software Qualities

In software engineering domain the accuracy and suitability of an architecture is analyzed considering several quality attributes. Bass (1998) and Clements (2002) propose the following Usability, Performance, Reliability, Availability, Security, Functionality, Modifiability, Portability, Variability, Subsetability, Testability, Conceptual Integrity, Building simplicity, Cost and Time to market.

The SA qualities are interrelated, and enhancing one will likely degrade or enhance others – for instance performance is likely to degrade scalability – for further detail on this subject please refer to Gillies (1992) or Khaddaj and Horgan (2004).

Considering the software qualities that the stakeholders would like to evaluate and the SA design stage (among other factors), different software evaluation approaches might be used.

3.2. Software Evaluation Approaches

The main aim of an evaluation is to analyze the architecture in order to identify potential risks and verify that the quality requirements have been addressed in the design (Dobrica and Niemela 2002).

In SA is recognized that it is not possible to measure the quality attributes of the final system based on SA design (Bosch and Molin 1999). This would imply that the detailed design and implementation represent a strict projection of the architecture. The aim of analyzing the architecture of a software system is to predict the quality of a system before it has been built and not to establish precise estimates but the principal effects of the architecture (Kazman et al. 1993).

Table 1 presents some architecture evaluation approaches.

Table 1. SA Evaluation Approaches

	SAAM	ATAM	ALMA	ARID	SBAR
Methods' Activities	6 activities	9 activities in 4 phases	5 activities carried out sequentially	9 activities in 2 phases	3 activities carried out iteratively
Methods' Goals	Risk Identification Suitability analysis	Sensitivity & Trade-off analysis	Change impact analysis, predicting maintenance effort	Validating design's viability for insights	Evaluate ability of SA to achieve quality attributes
Quality attributes	Mainly Modifiability	Multiple attributes	Maintainability	Suitability of the designs	Multiple attributes

A detailed classification and assessment of SA evaluation approaches is available in Babar et al (2004) and in Dobrica and Niemela (2002).

The ATAM (Architecture Trade-Off Analysis Method) evaluation approach, based on its predecessor SAAM (Software Architecture Analysis Method), evaluates an architecture considering multiple quality attributes (as modificability, performance, availability, security), providing orientation regarding attribute interdependencies and identifying architecture trade-off points (Kazman et al. 1998).

ATAM has 9 major steps, organized in 4 phases: Presentation (present ATAM, business drivers, and architecture), Investigation and Analysis (identify architectural approaches, generate quality attribute

utility tree, and analyze architectural approaches), Testing (brainstorm and prioritize scenarios, and analyze architectural approaches) and Reporting

One of ATAM most important steps is the generation of a quality attribute utility tree. The quality attributes that comprise system "utility" (performance, availability, security, modifiability, usability, and so on) are elicited specified down to the level of scenarios. An example of an ATAM utility tree is shown in Figure 1 (Clements et al. 2002).

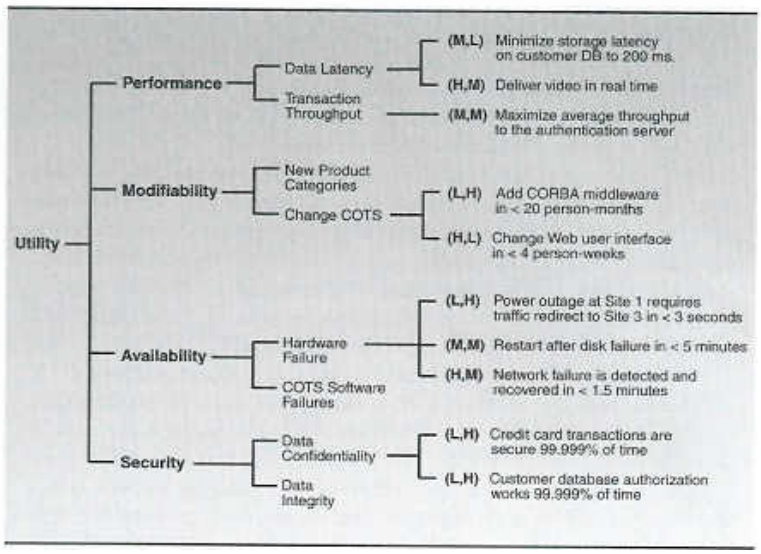

Figure 1. Sample Utility Tree (figure from Clements et al. 2002)

"Utility" is the root node (father of all the quality attributes); the second level represents the quality attributes (as performance, modifiability, availability and security); the third level is a refinement of the quality attributes (as hardware failure and COTS Software failures for availability, or data confidentially and data integrity for security); the leaves of the utility tree are concrete scenarios, that are then prioritized along two dimensions: importance of each scenario to the success of the system, and by the degree of difficulty posed by the achievement of the scenario (letters H,M,L next to scenarios description in Figure 1).

For further detail on the rest of ATAM approach please refer to Clements et al. (2002) book.

3.3. Software Metrics

According to Tom DeMarco "You cannot control what you cannot measure"; in software engineering, software metrics aim is to provide information to support quantitative decision-making during the software lifecycle, in order to "measure" your architecture (and control it!).

Over the last years, quite a few software metrics have been defined.

In order to measure software functionality, Albrecht and Gaffney (1983) proposed a Function Points metric (FP). The FP is computed by considering the inputs/outputs and the internally handled data (e.g., transactions from other applications, reports and message to the user or other application).

Functions Points are widely spread and have been used as a measure for several quality attributes, including size, productivity, complexity and functionality.

The number of Lines Of Code (LOC) is the oldest and most widely used measure of size. Since this metric is easy to understand and to measure has quite success and has been also used in other predictive models in terms of effort, fault-proneness, etc..

McCabe (1976), considering that the higher the number of paths in a program, the higher its control flow complexity probably will be, proposed the Cyclomatic Complexity metric. McCabe metric counts the paths from the start point to the end point of the program whose linear combinations provide all the paths in the program. Based on graph theory, the number of base paths in a program is computed as $v(G) = e - n + 2$, where e and n are the number of edges and nodes in the control glow graph, respectively.

With the emerging of the Objected Oriented (OO) paradigm, new concepts and abstractions appear such as classes, methods, messages, inheritance, polymorphism, overloading and encapsulation, which were not addressed in previous metrics. Chidamber and Kemerer (1995) and Basili (1996) proposed and tested a set of software metrics for OO development. These metrics are:

- Weighted Methods per Class (WMC), measures the complexity of an individual class.
- Depth of Inheritance Tree of a class (DIT).
- Number Of Children of a Class (NOC).
- Coupling Between Object classes (CBO) – A class is coupled to another one if it uses its member functions and/or instance variables.
- Response For a Class (RFC), is the number of methods that can potentially be executed in response to a message received.
- Lack of Cohesion on Methods (LCOM) is computed by subtracting the number of pairs of member functions with shared instance variables to the number of pairs of member functions without shared instance variables.

Several other metrics on OO development exist – such as Average dimension of methods, Average number of methods per class, Number of executable statements, Number of classes, Total number of methods, Number of times the method is inherited, Number of times the method is overloaded, medium time between (consecutive) failures (for further OO metrics see Abreu and Carapuça (1994) or Briand et al. (1998)).

4. Information System Architecture Evaluation

In this section, we start by presenting our Enterprise modelling framework, focusing at Information System level, and introducing ISA major primitives. Next we compare SA and ISA evaluation approaches, describe its similarities and differences and discuss what SA evaluation issues may be extended to ISA evaluation. Finally we propose a set of ISA evaluation metrics (mostly) based on the software engineering approach.

4.1. The CEO Modelling Framework

The CEO Framework aims at providing a formal way of describing business goals, processes, resources and information systems and the dependencies between them. It is composed of three separate levels, each of which provides adequate forms of representing the notions about the layer being described (Vasconcelos et al. 2001).

The modelling language used to implement the CEO Framework was UML (Unified Modelling Language) – for further reading, refer to Vasconcelos et al. (2001). Figure 2 presents the UML metamodel defined for the CEO Framework.

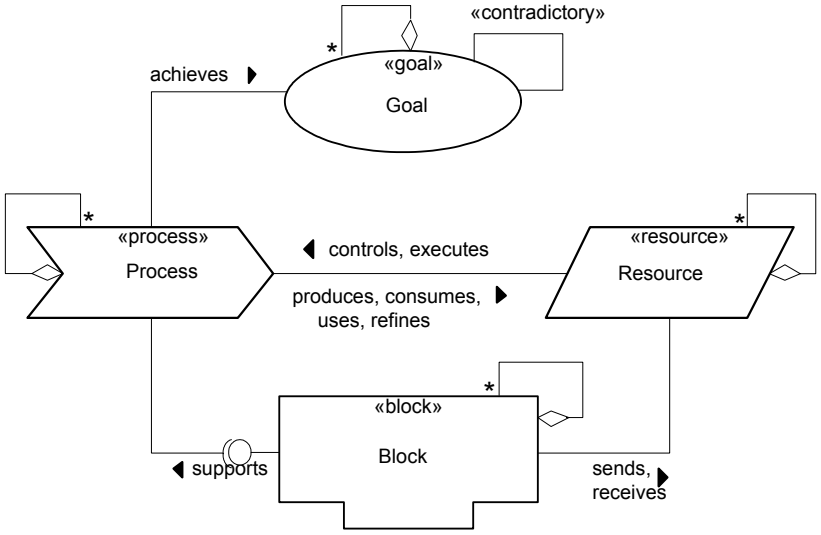

Figure 2 - UML Metamodel of the CEO Framework (Vasconcelos et al. 2001)

In order to model ISA key concepts the «Block» component was specialized. The key concepts for the ISA are:

- Information Entity – person, place, physical thing or concept that is relevant in the business context;
- IS Block – the collection of mechanisms and operations organized in order to manipulate organization data.
- IT Block – the infrastructure, application platform and technological/software component that realizes (or implement) an (or several) IS Block(s). IT Block defines three major sub-concepts:
    - IT Infrastructure Block – represents the physical and infra-structural concepts existing in an ISA: the computational nodes (as servers, personal computers or mobile devices) and the non-computational nodes (as printers, network, etc.) that support application platforms.
    - IT Platform Block – stands for the implementation of the services used in the IT application deployment.
    - IT Application Block, the technological implementation of an IS Block.
- Service – is an aggregation of a set of operations provided by an architectural block. A generalization of the web service notion (W3C 2002). We consider three distinct services in an ISA:
    - Business Service, collection of operations provided by IS Blocks that support business processes;
    - IS Service, set of operations provided by an IS Block to others IS Blocks;
    - IT Service, technological services provided by application platforms (Open 2001).
- Operation, the abstract description of an action supported by a service (the minor level concept in an ISA).

Figure 3 describes how these high-level primitives are related, in a UML profile for ISA. For further detail please refer to Vasconcelos et al (2003).

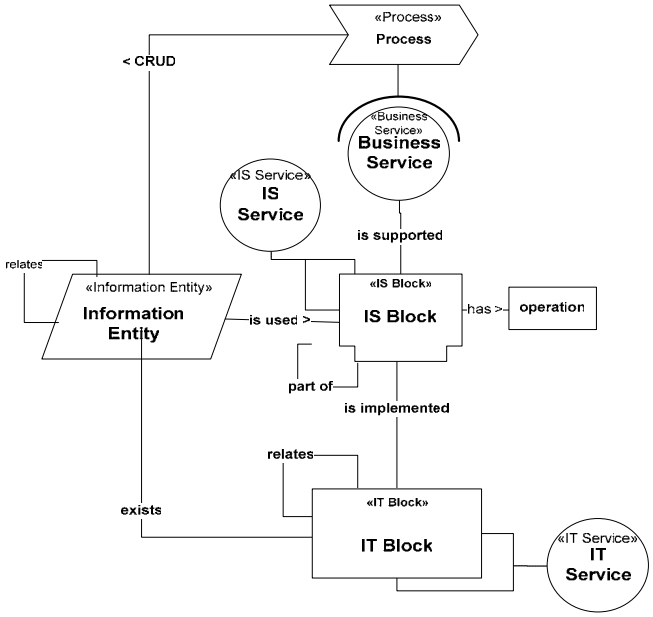

Figure 3. Information System CEO framework metamodel (Vasconcelos et al. 2003)

In order to describe these ISA primitives we are currently working on consolidating a set of attributes (the architectural attributes), that will characterize the properties of the Information System – for example, at ISA integration level, we have proposed a set of architectural attributes in Vasconcelos et al (2004b).

4.2. Information System Architecture and Software Architecture Evaluation

As presented in section 3.2, SA evaluation main goal is to analyze the architecture in order to identify potential risks and verify that the quality requirements have been addressed in the design, predicting the quality of the system before it has been built and establishing the principal effects of the architecture.

These are also ISA evaluation major goals. The main difference is that the focus is changed from a single software system (in SA evaluation) to multiple Information Systems (at application, information and technology levels) that should support the enterprise business processes (in ISA evaluation).

The fact that SA and ISA evaluation have similar goals, as discussed previous, motivate the authors to explore the applicability of SA qualities, metrics and evaluation approaches to ISA level. Our preliminary findings, discussed in the following subsections, are that:

- The quality attributes that are relevant for SA evaluation are generally applicable (and relevant) for ISA evaluation, such as performance, reliability, availability, modifiability, portability, security, etc.;
- The SA evaluation approaches (namely ATAM) are reasonably independent of the software engineering domain, thus theirs steps might be used for ISA evaluation;
- SA metrics are not applicable to ISA evaluation, since SA metrics deal directly with software attributes (as classes, lines of code, variables, etc.).

4.2.1. Qualities

Software engineering domain has a set of qualities that are commonly used when evaluating a software program (such as the ones described in section 3.1). In the information system domain (specifically in the enterprise information system architecture) this “consensus” does not exist, mostly

because the research on this subject is younger (and as a consequence of some confusion, until recently, between the information system and software engineering domains, in the authors' opinion).

Nevertheless, in the information system technological (or technical) architecture the Khaddaj and Horgan (2004) article presents an effort to identify quality attributes for information systems (still from a software point of view), such as performance, scalability, cost/benefit, usability, portability, robustness, correctness, and reliability. TOGAF framework (Open 2003) also presents an important research concerning information system qualities from a technical view point, namely: availability, manageability, performance, reliability, recoverability, locatability, security, integrity, credibility, usability, adaptability, interoperability, scalability, portability, and extensibility.

If we compare these IS technical qualities to software qualities (presented in section 3.1), we can conclude that the qualities attributes that are important in SA evaluation are also significant in ISA evaluation.

Although that at information system technological level the software qualities are applicable for ISA evaluation, the application and information sub-architectures (and its relations to business level) are not directly addressed by software qualities. For instance: which quality attributes are pertinent to assess the business support and alignment (in an ISA)? Or, which qualities are relevant in order to verify the alignment between the Information System strategy and the ISA?

4.2.2. Metrics

As described before, metrics are quantitatively interpretation of the observable architectural attributes. Since the architectural attributes are distinct in information systems and software, the ISA metrics and SA metrics can not be the same.

As presented in section 4.1, at ISA level the key concepts (primitives) relevant for application, information and technological architectures – such as «IS Block», «Information Entity», «IT Block», «Service», «Business Service», «IT Infrastructure Block», «Server» – are divergent from the SA attributes, more focuses on how programs or application components are internally built in order to implement a software program – such as lines of code, objects, classes, methods, variables, software algorithms, etc.

Therefore, despite the fact that the key ISA qualities (or requirements) are similar to SA qualities (such as performance, reliability, performance, etc.), the ISA metrics and SA metrics are distinct.

Thus software metrics such as the Number of Lines Of Code or the McCabe's Cyclomatic Complexity metric are not useful at ISA level (since the lines of code or the program algorithm are not enterprise information system attributes). However the authors believe that some of the SA metrics might be "extend/adapted" to the ISA domain (this issue is discussed in section 4.3).

Some metrics have already been proposed for evaluating the characteristics of an ISA, namely:

- For the information/data architecture there are some metrics focused on entity-relationship (ER) diagrams – for further detail see Gray et al. (1991), Kesh (1995), Moody (1998) and Si-Said Cherfi et al. (2002) researches. Genero et al. (2003), considering previous researches presents a set of metrics for measuring the structural complexity of entity-relationship (ER) diagrams (e.g., as the number of entities within an ER diagram, number of composite attributes, number of relationships within an ER diagram, Number of Binary Relationships, among others);
- The quantification of alignment between the architectural levels is also an issue that has few metrics and little experienced research. Pereira and Sousa (2003), based on some practical observations and on a literature review, propose a set of metrics to analyse misalignment between business, information and application architectures, by counting: the information entities created only by one process (versus the total number of entities), the processes that create,

update and/or delete at least one information entity (versus the total number of processes) and the information entities read by at least one process (versus the total number of entities).

- The analysis of the alignment between an information system architecture and a reference ISA is also an issue that does not have much experienced research. Vasconcelos et al. (2004) propose a set of metrics for verification of the alignment between the Portuguese Healthcare Information System reference model and new information system projects – the metrics included: Functional Overlapping indicator (which considers functions implemented by the proposed project and the ones that already exist in other systems in the organization), Technology change indicator (that considers the new technology introduced by the project that is not used in other existing IS of the organization), Informational entity model compatibility indicator, System overlapping indicator, Interface disregarding indicator, among others.

4.2.3. Approaches

The approaches previously presented for SA evaluation main aim is to predict the quality of a system before it has been built and not to establish precise estimates but the principal effects of an architecture (Kazman et al. 1993). These approaches present a sequence of steps that guide the evaluation team. These steps, generally, are independent of the software or enterprise information system domain. The major decisions are left to the project team (or the architect) – such as defining which attributes are relevant (and how much), what scenarios to consider, identify the sensitive and trade-offs, etc. Thus, considering that ISA evaluation has similar goals, there aren't motives for not using the approaches presented for SA evaluation (namely ATAM) in ISA evaluation. Furthermore, using tested and mature evaluation approaches the authors believe that could improve the ISA evaluation results.

However, some steps of the SA evaluation approaches provide support in the evaluation process by presenting examples and patterns at software level. For instance, ATAM approach, in order to help the definition of scenarios, Clements et al. (2002) propose a characterization of the quality attributes (in terms of stimulus, environment and response)**Error! Reference source not found.**. In these points the SA evaluation approaches must be adapted for ISA evaluation.

4.3. Information System Architecture Evaluation Metrics

Next, considering that the major differences between software evaluation and IS evaluation are at the metrics level, we propose a set of ISA evaluation metrics based on the software engineering metrics (described in section 3.3).

The metrics presented in next table were developed over the set of the most used and significant software metrics. The authors' goal, in this paper, is not to prove that the proposed metrics are the most adequate for ISA evaluation, but to show a correspondence between software and information system evaluation concepts.

In Table 2 we describe the SA metric that motivates the proposed ISA metric ("SA metric" column), the proposed ISA metric ("ISA metric" and "ISA metric description" columns), the ISA sub-architectures (application, information nor technological architectures) that this metric is applied to ("ISA arch." column) and the ISA qualities that are assessed ("ISA qualities" column).

Table 2. Proposed ISA metrics (based on software metrics)

SA metric	ISA metric	ISA metric description	ISA arch.	ISA qualities
Lines of Code (LOC)	Number of Applications	Number of applications in an ISA	App. Arch.	Cost Time-to-market
Total Number of instance variables	Number of Information Entities	Number of information entities in a ISA	Inf. Arch.	Adaptability Modificability
Lack of COhesion in	Average Lack of	Average number of disjoint sets of	App. Arch.	Adaptability

SA metric	ISA metric	ISA metric description	ISA arch.	ISA qualities
Methods (LCOM)	COhesion in Application blocks	information entities used by the IS block operations in each application. Greater number of disjoint sets imply lack of cohesion of the application, so the application should be split into two or more application blocks.	Inf. Arch.	Modificability
Average number of methods per class	Average number of operations per application block	Average number of operations in application blocs («IS blocs»)	App. Arch.	Cost Adaptability Modificability
Cyclomatic Complexity	Average Service Cyclomatic Complexity	The Service Cyclomatic Complexity is computed by counting the number of possible paths needed to support a business service	App. Arch.	Cost Adaptability Modificability
Coupling Between Object classes (CBO)	Average Coupling Between Applications	Average number of IS services (provided by other applications - «IS Blocks») to which a application («IS Block») is coupled, i.e., those applications that use that application or are used by that application (consuming or providing «IS services»)	App. Arch.	Modificability Adaptability
Response For a Class (RFC)	Average Response For a Service	Average number of applications that can be used when a Service receives a request (business or application)	App. Arch.	Testability Modificability Security
System Medium Time Between (consecutive) Failures (MTBF)	Weight Service Medium Time Between (consecutive) Failures	Computed as the average Time Between (consecutive) Failures for each business service, considering the business importance (weight) of each service.	App. Arch. Tech. Arch.	Reliability Availability

In the previous table, by considering the enterprise information system concerns and attributes, and adapting software attributes inherent to the software metrics, we propose a set of ISA metrics. The metrics proposed are based on the most widely accepted and used software metrics.

Although adapting the software metrics to the ISA domain provides a more solid basis for ISA evaluation, it does not ensure that all the ISA concerns are consider in the evaluation. For instance, one of the most important issues in the ISA is the degree of business support and automation; thus a simple and important metric when analysing an ISA (and its support to business) is the ratio between the number of business services required by the business processes and the business services («Business Service») actually provided by the information systems («IS Block») – this metric is not directly derived from a software metric.

In order to evaluate an ISA one should used an evaluation approach; as described in section 4.2.3, the software evaluation approaches could be used for ISA evaluation; in the case study described in next section we will use some of the ATAM steps for an ISA evaluation.

5. An Information System Architecture Evaluation Case Study

In this section we present an ISA evaluation case study, in the Portuguese health-care industry. In this case study, our R&D unit was invited by Saúde XXI (the Portuguese office responsible for managing the European funds in the Public Health Care System) to evaluate a project proposal: the Call Doctor Project[1].

In this case study we applied some steps of ATAM software evaluation approach, and using CEO modelling framework (in order to represent the ISA) we also applied some of the metrics proposed in section 4.3.

5.1. The Call Doctor Project

In Portugal about 92% of the population has a mobile phone (there are about 9,6 million mobile users of about a 10 million people population) – (Anacom 2005). Considering these facts, a Portuguese company pretends to implement a system that would support the efficient connection between physicians (working in the Public health-care System) and theirs patients, using the mobile phone – we will call this project: The Call Doctor Project.

The Call Doctor Project pretends to provide each medical doctor a mobile phone/PDA with a private and a public phone numbers; the patient may call his/her medical doctor when he/she needs, paying an added price per second (than in a normal call), but having online and immediate health-care assistance; when the physician is not available the patient call is redirect to a centralized call-center that provides the basis guiding for the patient and manages the physician timetable (adding and removing patient appointments).

The physician will also have access to a drug database and, in the near future, the Call Doctor System pretends to be integrated with the Pharmacies Information Systems in order to allow online prescription. Another important issue in this project is the patient clinical data integration between the mobile phone, the central system and the Public health-care Information systems (in the hospitals and in the primary health-care units).

In this project our research center, considering our previous experience on the area and our previous cooperation with Saúde XXI (for example see Vasconcelos et al. 2004), was responsible by the technological evaluation of the project. In this paper we only highlight the most significant evaluation steps and metrics used in the case study.

5.2. The Evaluation Approach

In order to perform an evaluation of the Call Doctor Project, we put up an evaluation team composed by different stakeholders with complementary skills, namely two physicians, a financial consultant, and three technological consultants. This evaluation team follow ATAM major steps.

Before analysing further the project proposal, the team identified the system major requirements and quality attributes. Using ATAM methodology, the team generated an utility tree, presenting the system major qualities, specified down to the level of scenarios. Figure 4 presents an extract of the utility tree.

[1] The facts presented here stand for a hypothetical project proposal (all names, brands and facts, for confidentiality reasons, are fictitious); however, this case study is based on our experiences and participation on the evaluation of real IS/IT health care projects, where analogous proposals where evaluated.

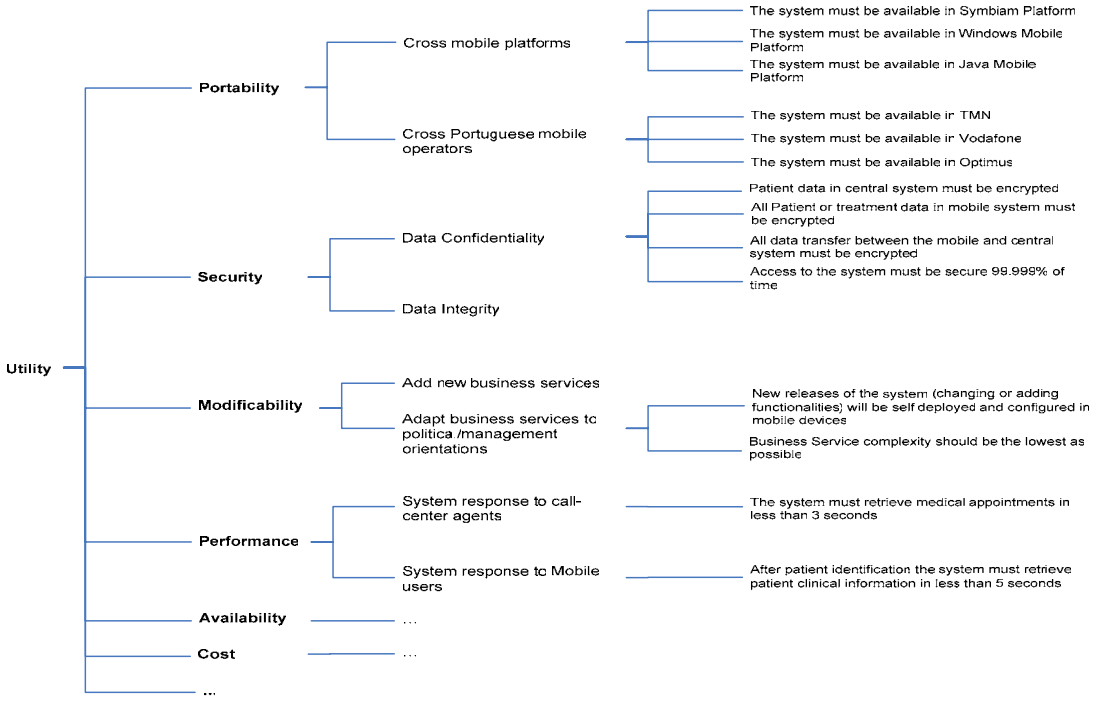

Figure 4. Call Doctor Utility Tree (incomplete)

In order to ensure the independence between the mobile part of the system and the mobile phone suppliers and the mobile Operators, the Portability was identified as an important system quality, as security, modificability, performance, availability and cost (among others).

5.3. Information System Architecture Assessment

The description of the Call Doctor Project was presented to the evaluation team in several plain text written pages. Thus, one of the preliminary tasks of the team was to transform those “word oriented” description of the architecture to a more engineering and “diagram oriented” description. The team adopted the CEO modelling profile to represent the ISA.

Simultaneously to the process of representing the ISA, the evaluation team used some of the metrics described previously and found some potential gaps in the project, that pointed out to the project proponent.

For instance, the team identified that the “Provide phone health-care” process required three business services (Physician appointment management, Patient clinical management, and drug prescription), however the mobile system description did not provided one of the business services (drug prescription) – as presented in Figure 5.

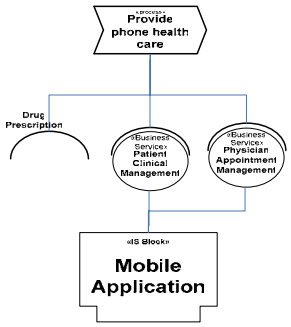

Figure 5. Business/Application architecture relations (functional misalignment)

In Figure 5 the ratio between the number of business services required by the business processes and the business services actually provided by the application blocks metric is $\frac{2}{3}$, since the Drug Prescription did not have the correspondent business service. This gap was reported to the project proponent that corrected the project proposal. The global ISA, at application level (after this short correction) is presented in Figure 6.

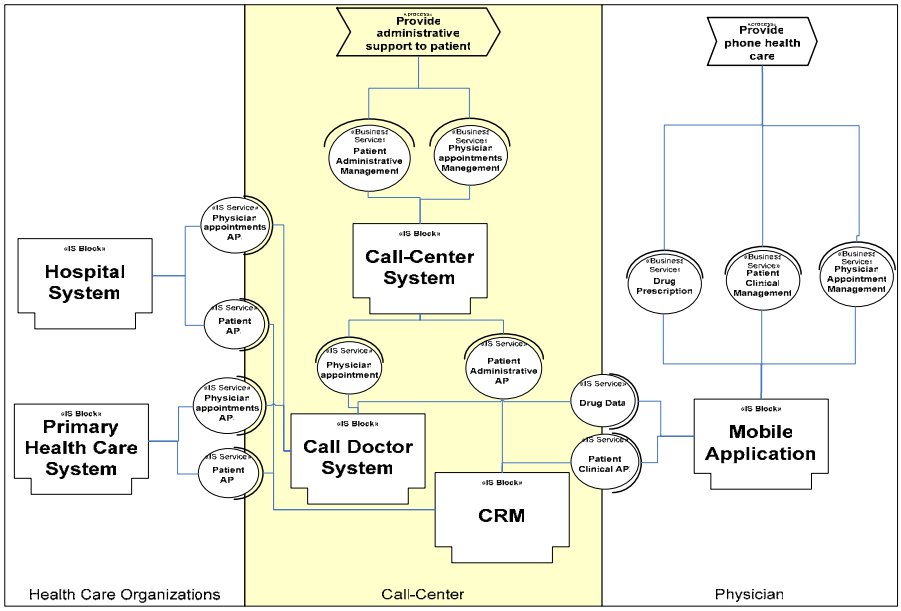

Figure 6. Call Doctor Application Architecture

A simple metric to estimate time-to-market by comparing with other ISA is counting the number of applications. In this ISA we have 4 applications (plus two existing ones in the health-care organizations).

In order to verify the performance scenario described in Figure 4 ("system response for mobile users"), the team detailed the business service (presenting its operations) - Figure 7**Error! Reference source not found.**. In this figure we can see that the business operations of the service are: "Get Patient Record", "Update Patient Information", and "Search Patients".

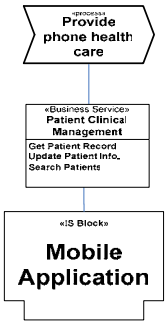

Figure 7. Patient clinical management business service

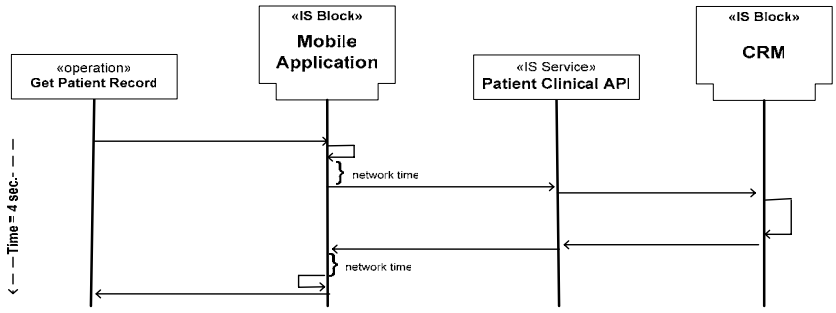

Figure 8. Patient clinical management business service

For the "Get Patient Record" operation the sequence of applications (and services) and messages of a possible scenario are described in Figure 8**Error! Reference source not found.**.

In the previous figure the total time of the operation Get Patient Record is 4 seconds; however this scenario describes an average situation, since, in worst case scenarios, the communications might be

slower, or the patient record might not exist in the CRM application and it might be necessary to invoke the services provided by the hospital or primary health-care unit.

Figure 9 presents the behaviour diagram for the patient clinical management business service. From this diagram we can compute this service cyclomatic Complexity, which is: *v(G) =e-n*+2 = 8-5+2=5.

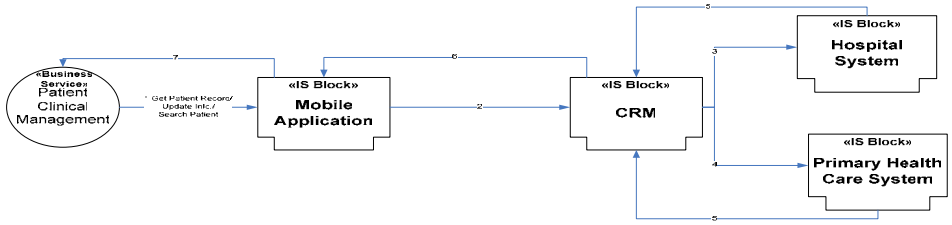

Figure 9. Application architecture Behaviour Diagram (Patient Clinical Management service)

The response for the Patient Clinical Management Service metric is 4, since four applications might be used to support this service.

The evaluation team aided by these (and other) metrics and supported on ATAM approach presented a set of architectural recommendations in order to better accommodate the qualities identified (in the utility tree) – for example ensure the deploy of the mobile application in the major mobile platforms (Symbian, Windows Mobile and Java Mobile).

Figure 10 describes the technological architecture of the Call Doctor Information System, after the recommendations of the evaluation team.

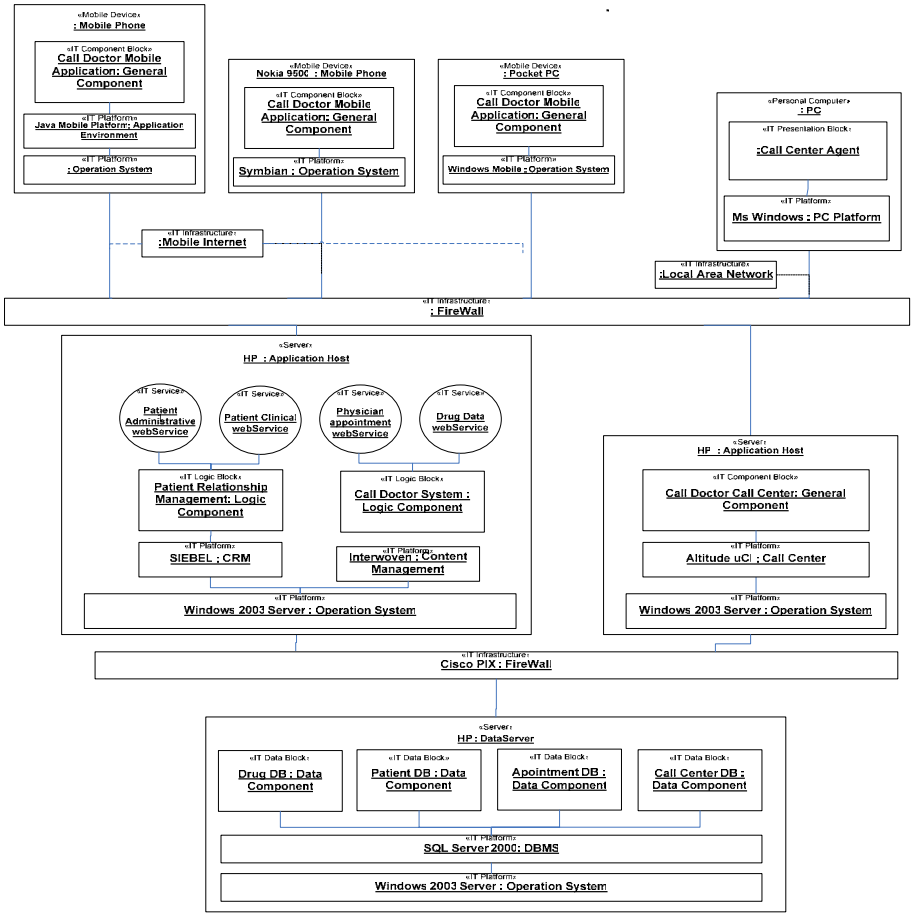

Figure 10. Call Doctor Technological architecture

6. Conclusions and Future Work

This paper provides an overview on the software evaluation approaches, qualities and metrics and theirs relevance for the enterprise information system architecture evaluation.

In this paper the authors conclude that the quality attributes relevant for SA evaluation are generally applicable for ISA evaluation (such as performance, reliability, availability, Modifiability, portability, security, etc), that the SA evaluation approaches (namely ATAM) might be used for ISA evaluation (since they are reasonably independent of the software engineering domain), and that the SA metrics are not applicable to ISA evaluation (since SA metrics deal directly with software attributes that do not have any meaning in the ISA context).

Considering the similarities between the software engineering and the information system engineering domains, the authors propose several ISA metrics (based on experienced SA metrics), namely: Number of Applications, Number of Information Entities, Average Lack of COhesion in Application blocks, Average number of operations per application block, Average Service Cyclomatic Complexity, Average Coupling Between Applications, Average Response For a Service, and Weight Service Medium Time Between (consecutive) Failures.

The health-care case study supports these findings and proposals, namely because the evaluation team used the ATAM approach for evaluating an ISA, and some of the proposed metrics provided a guidance in the evaluation process.

Nevertheless, the authors recognize the immaturity of the proposed metrics, and that further research and experimentation on this topic is required (in order to clarify the metrics impact on all the quality attributes).

As future work, we are working on a computer tool that supports the modelling of ISA, using the CEO modelling UML profile, and displaying the metrics – for guidance on the architecture definition and assessment.

7. Acknowledgements

The research presented in this paper was possible thanks to the support of Saúde XXI.

8. References

Abreu, F., and R. Carapuca, `*Candidate metrics for object-oriented software within a taxonomy framework*, J. Syst. and Software 23, 87-96, 1994.

Babar, M, L. Zhu, and R. Jeffery, *Framework for Classifying Sofwtare Architecture Evaluation Methods, Australian Software Engineering Conference (ASWEC'04),* 2004

Clements,P., R. Kazman, and M.Klein, *Evaluating Software Architectures: Methods and Case Studies*, Addison-Wesley Professional , ISBN 02017048, 2002

Dobrica, L., and Niemela, E., *A Survey on Software Architecture Analysis Methods*, IEEE Tansactions on Software Engineering, vol. 28, no. 7, 2002.

Genero, M., G. Poels, and M. Piattini, Defining and Validating Metrics for Assessing the Maintainability of Entity-Relationship Diagrams, Working Paper, Faculteit Economie En Bedrijfskunde, 2003.

Kazman, R., J. Asundi, and M. Klein, *Making_Architecture_Design_Decisions_An Economic Approach*, Technical Report, Carnegie Mellon Software Engineering Institute, 2002

Maes, R., D. Rijsenbrij, O. Truijens, and H. Goedvolk, *Redefining Business – IT Alignment Through a Unified Framework*, White Paper, May 2000.http://www.cs.vu.nl/~daan/

McCabe, T. J. ``A Complexity Measure," IEEE Trans. Software Eng. 2, 1976

Open Group, *The Open Group Architectural Framework (TOGAF) – Version 8.1*, The Open Group, December 2003.

Pereira, C., and P. Sousa, *Getting Into The Misalignment Between Business And Information Systems*, 10th European Conference On Information Technology Evaluation, 2003.

Si-Said Cherfi S.. Akoka J. and Comyn-Wattiau I, *Conceptual Modelling Quality – from EER to UML Schemas Evaluation*. 21st International Conference on Conceptual Modeling (ER 2002). Tampere. Finland. 499-512, 2002.

Spewak, S., and S. Hill, *Enterprise Architecture Planning: Developing a Blueprint for Data, Applications and Technology*, Wiley-QED, ISBN 0-471-599859, 1992

Vasconcelos, A., A. Caetano, J. Neves, P. Sinogas, R. Mendes, e J. Tribolet, *A Framework for Modeling Strategy, Business Processes and Information Systems*, 5th International Enterprise Distributed Object Computing Conference EDOC, Seatle, USA, 2001.

Vasconcelos, A., C. Pereira, P. Sousa and J. Tribolet, *Open Issues On Information System Architecture Research Domain: The Vision*, Proceedings of the 6th International Conference on Enterprise Information Systems (ICEIS 2004), Portugal, 2004c.

Vasconcelos, A., Miguel Silva, António Fernandes, and José Tribolet (b), An Information System Architectural Framework for Enterprise Application Integration, Proceedings of 37th Annual Hawaii International Conference On System Sciences (HICCS37), Hawaii, USA, 2004.

Vasconcelos, A., P. Sousa, and J. Tribolet, “Information System Architectures”, Proceedings of Business Excelence 2003, International Conference on Performance Measures, Benchmarking and Best Practices in New Economy, Portugal, 2003.

Vasconcelos, A., R. Mendes, and J. Tribolet, *Using Organizational Modeling to Evaluate Health Care IS/IT Projects*, Proceedings of 37th Annual Hawaii International Conference On System Sciences (HICCS37), Hawaii, USA, 2004.

Full list of references available in http://ceo.inesc.pt/andre/papers/ecite05/referencesISAEval.doc