FEDERAL UNIVERSITY OF SERGIPE

CENTER OF EXACT SCIENCES AND TECHNOLOGY

POSTGRADUATE PROGRAM IN COMPUTER SCIENCE

Design and Evaluation of an Architecture Conceptualization Framework System based on ISO/IEC/IEEE 42020

Master Thesis

Valdicélio Mendes Santos

São Cristóvão – Sergipe

2023

FEDERAL UNIVERSITY OF SERGIPE

CENTER OF EXACT SCIENCES AND TECHNOLOGY

POSTGRADUATE PROGRAM IN COMPUTER SCIENCE

Valdicélio Mendes Santos

Design and Evaluation of an Architecture Conceptualization Framework System based on ISO/IEC/IEEE 42020

Master Thesis presented to the Post-Graduation Program in Computer Science of the Federal University of Sergipe as a partial requirement to obtain a master's degree in Computer Science.

Advisor: Michel dos Santos Soares

São Cristóvão – Sergipe

2023

I dedicate this work to God, my family and my advisor and to all that helped me to accomplish it.

Acknowledgements

I praise and thank the Holy Trinity and Our Lady for life as a child of God. Life is seeking the Kingdom of God and His Justice, with love and for love, and everything else will come in addition. The Lord leaves us lacking nothing. Gratitude to my grandparents (in memoriam), my parents Vera Lúcia and José Guimarães, for their natural life, unconditional love and education. The best inheritance from our parents is Faith and Education. To my siblings, Joseane and Uberlan, for the walk and sharing since the origins. To my beloved wife, Lícia Violeta, treasure that the Lord granted me to live in the Love of the sacramental union that leads to sanctity. Thank you for your unconditional support, patience and incentive. Our supervisor is someone we esteem, admire and who inspires us. Prof. Michel, thank you for your patience, stimulus, support and friendship. With your words, the challenges seem to be simpler. To the dear teachers and collaborators with DCOMP for every teaching, help and encouragement. You are not only excellent professionals, but human beings with a big heart. To the brothers on the journey of faith, colleagues and friends of work and studies that I have done so far, regardless of the distances, know that you can count on.

Resumo

Do ponto de vista da Engenharia de Software, entre as dificuldades para desenvolver *software-intensive systems* estão a necessidade de gerenciar e controlar dados que devem ser mantidos por décadas, mesmo considerando a evolução da tecnologia nos anos seguintes, bem como a necessidade de cooperar com sistemas legados e descrever as necessidades e *concerns* de uma variedade de *stakeholders*. O desenvolvimento de *software-intensive systems* baseado em uma sólida arquitetura de software é um fator de sucesso que não pode ser negligenciado. Entretanto, os processos relacionados à arquitetura de *software-intensive systems* são freqüentemente considerados apenas a partir de um baixo nível de abstração, mesmo para uma descrição da arquitetura de software. Uma norma recente de arquitetura, a ISO/IEC/IEEE 42020, define 6 Cláusulas para o processo de arquitetura, entre elas o processo de Conceitualização da Arquitetura, que é o tema deste estudo. Dada a importância de estabelecer uma arquitetura de software bem definida, considerando as dificuldades de compreensão de uma norma arquitetural, e considerando também que a ISO/IEC/IEEE 42020 só recentemente foi publicada, este trabalho propõe um *framework* e, em seguida, o projeto e avaliação de uma aplicação a ser executada em ambiente web para apoiar os arquitetos de software na utilização das atividades e tarefas da cláusula de Conceitualização de Arquitetura baseada no *framework* descrito. O ArchConcept foi projetado para abordar a abstração de alto nível da norma ISO/IEC/IEEE 42020, e pode ser útil para arquitetos de software que desejam seguir a recomendação da ISO/IEC/IEEE 42020 e alcançar resultados de alta qualidade em seu trabalho de conceitualização da arquitetura de software. Uma avaliação qualitativa por meio de um questionário foi realizada a fim de obter informações sobre as percepções dos profissionais sobre o ArchConcept, de acordo com o Modelo de Aceitação de Tecnologia (TAM). Como o ArchConcept se concentra nos estágios iniciais do projeto (Conceitualização da Arquitetura), os resultados encontrados neste trabalho poderiam ser uma evidência do pouco tempo dedicado à fase inicial dos projetos e suas conseqüências, como mal-entendidos. Alguns respondentes não perceberam a utilidade do sistema ArchConcept porque não perceberam a importância da Conceitualização da Arquitetura para o sistema, ou porque há uma falta de conhecimento, pois 13 respondentes estavam interessados em saber mais sobre o processo de Conceitualização da Arquitetura. Como a Norma ISO/IEC/IEEE 42020 é nova e ainda não é bem conhecida na indústria e academia, os resultados indicaram que o ArchConcept não foi devidamente percebido em termos de utilidade e uso, embora muitos entrevistados tenham se dado conta disso apenas lendo o guia do usuário. Por outro lado, a aplicação foi considerada fácil para o uso diário.

Palavras-chave: Arquitetura de Software, ISO/IEC/IEEE 42020, Conceitualização da Arquitetura, Modelo de Aceitação de Tecnologia.

Abstract

From the Software Engineering point of view, among the difficulties for developing software-intensive systems are the necessity of managing and controlling data that must be held for decades, even considering the evolution of technology in the following years, as well as the necessity of cooperating with legacy systems and describing the needs and concerns of a variety of stakeholders. Developing software-intensive systems based on solid software architecture is a success factor that cannot be neglected. However, the processes related to the software architecture of software-intensive systems are often considered only from a low level of abstraction, even for the description of the software architecture. A recent architectural Standard, the ISO/IEC/IEEE 42020, defines 6 clauses for the architecture process, among them the Architecture Conceptualization process is the subject of this study. Given the importance of establishing a well-defined software architecture, considering the difficulties of understanding an architectural Standard, and also considering that ISO/IEC/IEEE 42020 has only recently been published, this work proposes a framework, and then the design and further evaluation of a web-based application to support software architects in using the activities and tasks of the Architecture Conceptualization clause based on the framework described. The ArchConcept was designed to address the high-level abstraction of the Standard ISO/IEC/IEEE 42020, and can be useful for software architects that want to follow ISO/IEC/IEEE 42020's recommendation and achieve high-quality results in their work of software architecture conceptualization. A qualitative evaluation by means of a questionnaire was carried out in order to obtain information about the perceptions of professionals regarding the ArchConcept, according to the Technology Acceptance Model (TAM). As ArchConcept is focused on the early stages of the project (Architecture Conceptualization), the results found in this work could be an evidence of the short time dedicated to the initial phase of projects and their consequences, like misunderstandings. Some respondents answered they do not perceive the usefulness of the ArchConcept system because they did not realize the importance of the Architecture Conceptualization to the system, or because there is a lack of knowledge, as 13 respondents were interested in knowing more about the Architecture Conceptualization process. As the ISO/IEC/IEEE 42020 Standard is new and still not well-known in industry and academia, the results indicated that the ArchConcept was not properly perceived in terms of usefulness and usage, although many respondents realized it just by reading the user guide. On the other hand, the application was considered easy for daily use.

Keywords: Software architecture. ISO/IEC/IEEE 42020. Architecture Conceptualization. Technology Acceptance Model.

List of Figures

List of Tables

List of abbreviations and acronyms

DCOMP	Departamento de Computação
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
ISO	International Standards Organization
UFS	Universidade Federal de Sergipe

Contents

1 Introduction

Software architecture has been defined in various classic ways. It has been defined as a structure composed of components, and rules characterizing the interaction of these components (JONES, 1994). It has been defined as components, connections, constraints and rationale (BOEHM, 1993). It has also been defined as elements, form, and rationale (PERRY; WOLF, 1992); and as components, connectors, and configurations (GARLAN; SHAW, 1993). These definitions appear to focus on the architectural representation, but it is not clear that they fully address the full range of evaluation issues associated with software architecture, which means, a set of views, decisions and elements which describe the architecture. Nevertheless, Boehm et al. (1995) states that a software system architecture comprises: a collection of software and system components, connections, and constraints; a collection of system stakeholders' need statements; and a rationale that demonstrates that the components, connections, and constraints define a system that, if implemented, would satisfy the collection of system stakeholders' need statements. This last definition suits this work because software architecture can be seen as a set of processes that meet stakeholders' needs.

Software architecture is the fundamental organization of a system, embodied in its components, their relationships to each other and the environment, and the principles governing its design and evolution (ANSI/IEEE, 2000). Software architecture is an important area of Software Engineering as it is used to manage the development and maintenance of large-scale software systems (FALESSI et al., 2010), and many researchers and practitioners have recognized it as a key asset in the software life cycle (GARLAN, 2000; KRUCHTEN; OBBINK; STAFFORD, 2006; GARLAN, 2014).

Software architecture is a high-level design and is used as a basis for detailed design. The basis for software design is better represented by a software architecture, which should provide a layered approach. In simple terms, the description of the architecture of a software system involves the elements that compose the system and how they interact, including the interaction points. Software architecture is part of the design, but with a focus on significant elements, such as major classes and their communications, patterns, frameworks, layers, subsystems, components and interfaces (KRUCHTEN, 2003). The way these elements are distributed, communicate with each other and with the system environment is also described in the software architecture. A common misconception is that software architecture is only about the structure of elements and their relationships (KRUCHTEN, 2003). Software architecture is also about behavior and interfaces between elements.

Design for software-intensive systems requires adequate methodology in order to support the development of these systems (TIAKO, 2008). However, the theory of modeling for software-intensive systems remains incomplete, and methodologies for specifying and verifying software-intensive systems pose a grand challenge that a broad stream of research must address (BROY, 2006).

In large, complex systems, it is common to have a hierarchy of requirements, and their organization into various levels helps in dealing with system complexity. For instance, high-level business requirements may be gradually decomposed into more detailed software requirements, forming a hierarchy.

A variety of processes, methods, languages, frameworks and architectures have been proposed to deal with the challenge of developing software-intensive systems. Activities related to each one of these software engineering elements have been proposed in the past decades. However, processes related to software architecture are still often neglected (BOOCH, 2007; KNODEL; NAAB, 2014; WOODS, 2016; GALSTER; TAMBURRI; KAZMAN, 2017; DASANAYAKE et al., 2019), even though it is well-known in industry and academia that software architecture processes are considered a critical success factor (BROWN; MCDERMID, 2007; GARLAN, 2014; JUNIOR; MISRA; SOARES, 2019). Although there are debates on the role of describing software architectures in agile software development processes (PACE et al., 2016; HODA; SALLEH; GRUNDY, 2018; ULUDAg; MATTHES, 2020), there are important results that consider software architecture as a crucial technical component for agile development (ABRAHAMSSON; BABAR; KRUCHTEN, 2010; YANG; LIANG; AVGERIOU, 2016), even though there is still a lack of description and analysis regarding the costs and failure stories of the combination of agile and architecture-based software development processes.

Research on software architecture is rapidly growing as many researchers and practitioners recognize the importance of having a well-defined software architecture to support activities such as project management, design and implementation. On the other hand, poor software architecture definition is recognized as a major technical risk when designing a software-intensive system (CLEMENTS et al., 2002). In summary, software architecture is a critical element for developing software-intensive systems (SOARES, 2010).

1.1 Contextualization

The complexity of human-made systems has grown to an unprecedented level, which leads to new opportunities and greater challenges for organizations that create, trade and utilize systems. To address these opportunities and challenges, it is increasingly necessary to apply concepts, principles, procedures and tools to make better architecture-related decisions, more effective architectures, better architecture strategy and increased architecture maturity. Architecture-related activities are now strategic aspects of projects and enterprises, and the use of architecture frameworks has become common practice in commercial, government, civil and military domains (ISO/IEC/IEEE, 2019a).

The architecture shows, at an abstract level, the whole picture, and helps stakeholders to understand the consequences of the decisions that are taken prior to investment in the design and development of the elements of the system. The architecture shows how the many subsystems and components are interconnected, how they cooperate, and the boundaries and interfaces between each other. This is useful, for instance, for vendors creating their products according to proposed standards. In addition, the architecture enables the identification of major risks and how to mitigate them. A better understanding of the whole is useful to create project plans with realistic budgets and duration (SOARES, 2010).

Nevertheless, processes related to the software architecture of software-intensive systems are often considered only from a low level of abstraction, even for a description of the software architecture. There is a need to increase the level of abstraction, hiding whenever possible unnecessary complexity by the intense use of models (BOOCH et al., 2007). Only relevant decisions are important at this level, i.e., those that have a high impact on the cost, reliability, maintainability, performance and resilience of the future system (SOARES, 2010).

Given the importance of Software Architecture processes, a family set of architecture Standards was proposed under reference number 420X0. The ISO/IEC/IEEE 42010 Standard (ISO/IEC/IEEE, 2011b) was proposed in 2011 with the main objective of being a recommended practice for creating a description of software architecture. Recently, a new Standard, the ISO/IEC/IEEE 42020 (ISO/IEC/IEEE, 2019a), was set in a document with the purpose of addressing a Standard for governance management, conceptualization, evaluation, and elaboration of architectures, and activities that enable these processes. Another Standard in the family, the ISO/IEC/IEEE 42030 (ISO/IEC/IEEE, 2019b), regards means to organize and record architecture evaluations for enterprise, systems and software fields of application. ISO/IEC/IEEE 42020 is the main subject of the study presented in this work.

The ISO/IEC/IEEE 42020 (ISO/IEC/IEEE, 2019a) was published by the Technical Committee ISO/IEC JTC 1/SC 7 Software and systems engineering which scope is the standardization of processes, supporting tools and supporting technologies for the engineering of software products and systems.

The importance of using International Standards such as ISO/IEC/IEEE 42020, given that it directly affects the quality and reliability of software, has been recognized by many authors (SANTOS; MISRA; SOARES, 2020; ASCHER; EHLERS, 2021; QUEZADA-GAIBOR et al., 2022). However, ISO/IEC/IEEE 42020 has only recently been considered in practice, and there is still a shortage of its actual use (SANTOS; MISRA; SOARES, 2020).

Each architectural process described in ISO/IEC/IEEE 42020 is organized in terms of purpose, desired outcomes, and a list of activities for achieving those outcomes. There are 6 architect processes in ISO/IEC/IEEE 42020: Architecture Governance, Architecture Management, Architecture Conceptualization, Architecture Evaluation, Architecture Elaboration and Architecture Enablement. This work addresses the process "Architecture Conceptualization", given that our purpose here is to understand stakeholders' needs, including their main concerns, requirements, purposes, processes, and objectives, before dealing with the architecture elaboration.

1.2 Research Problem

There are several classic definitions for Software Architecture, so there is no homogeneous consensus in the literature (KRUCHTEN; OBBINK; STAFFORD, 2006). However, it is known that Software Architecture is not the operational software, but rather, a representation that allows analyzing the effectiveness of the project to satisfy the declared requirements (PRESSMAN; MAXIM, 2014). Then, the architecture of a system is a set of decisions, so it eases or constrains the achievement of the desired system properties. Examples of decisions are the type of programming language to adopt, and the database for data persistence (FALESSI; AL, 2011).

The architecture shows, at an abstract level, the whole picture, and helps stakeholders to understand the consequences of the decisions that are taken prior to investment in the design and development of the elements of the system. The architecture shows how the many subsystems and components are interconnected, how they cooperate, and the boundaries and interfaces between each other. Future systems' maintenance and evolution are facilitated when the software architecture is clear for all stakeholders (LINDGREN; NORSTROM; WALL A. LAND, 2008). In addition, the architecture enables the identification of major risks and how to mitigate them. Finally, a better understanding of the whole is useful to create project plans with realistic budgets and duration.

Decisions made when defining the software architecture have a long-lasting impact on the design and major quality attributes of the software, such as performance, evolution and safety. These decisions will reflect on design models and on software implementation. As a matter of fact, the architecture description is an effective communication means between different stakeholders. A commonly accepted notion for developing software is that having a well-defined software architecture description is essential (KRUCHTEN; OBBINK; STAFFORD, 2006). Knowing the structure of the system and the relationship between components helps in software development and to communicate decisions to stakeholders.

Pareto et al. (2012) and Heesch, Avgeriou e Hilliard (2012) are a sample of how the activities of conceptualization of an architecture have been performed individually, if compared to the Conceptualization Process of the ISO/IEC/IEEE 42020 Standard, which proposes a set of interrelated tasks. This dissertation proposes to approach the 10 activities of the Conceptualization Process present in the ISO/IEC/IEEE 42020 in an unedited way, in order to evaluate its use and to highlight if there are advantages, according to the objectives established.

Most commonly, the works presented before addressed partially some of the activities and tasks of the architecture conceptualization process, such as stakeholder concerns, architectural decisions, problem space, solution space, and tradeoffs.

Some activities of the software architecture Conceptualization Process of the ISO/IEC/IEEE 42020 (ISO/IEC/IEEE, 2019a) were employed partially. One of the reasons for this situation is that there was no standard to discipline the activities and tasks of an architecture conceptualization. The need to use and evaluate the Conceptualization Process is justified since it is a current standard, but with a broad scope due to its list of activities and tasks.

1.3 Objectives

The main objective of this dissertation is to design and develop a framework and a related application to describe software architecture following the ISO/IEC/IEEE 42020 regarding the Architecture Conceptualization process, which presents high-level concepts and elements of software architecture. Given this proposal, first, it was identified the main aspects for the clause Architecture Conceptualization of ISO/IEC/IEEE 42020 and then proposed a framework that provides requirements to design and develop the web-based application.

The main objective of this dissertation is to evaluate the application of the Architecture Conceptualization Process proposed in the ISO/IEC/IEEE 42020 by means of the conceptualization of an Architecture for a web-based application to support software architects in using the activities and tasks of the Architecture Conceptualization.

The following specific objectives are identified:

- To propose a framework based on the activities and tasks of the Architecture Conceptualization Process described in the ISO/IEC/IEEE 42020 Standard.
- To develop a web-based application to assist in the management of the activities and tasks of the Architecture Conceptualization Process described in the ISO/IEC/IEEE 42020 Standard.
- To qualitatively evaluate the web-based tool developed.

1.4 Methodology

According to research classification, this master thesis presents an exploratory objective.

In terms of objectives, exploratory research aims to provide more familiarity with the subject, aiming to make it more explicit or to establish hypotheses (GIL, 2002). This type of research aims at the improvement of ideas or the discovery of intuitions. Thus, this research can be classified as exploratory because it proposes an Architecture Conceptualization Framework System based on the ISO/IEC/IEEE 42020 Standard.

As regards the technical procedures, Patton (2005) states that qualitative research implies three types of data collection: interviews, direct observations, and written documents. Interviews produce citations about people's experiences, opinions, feelings, and knowledge; data from observations consist of detailed descriptions of people's actions in the researched environment, and document analysis includes studying official publications, reports, and open-ended written responses to questionnaires and surveys (PATTON, 2005). According to Gil (2002), most of the time, research that has an exploratory feature ends up taking the form of bibliographic research or case study. A case study focuses on one instance of a phenomenon to be investigated, and it gives a detailed, in-depth description and insight of that instance. Additionally, complexity is fundamental to a case study to have success because it investigates multiple factors, events, and relationships that occur in a real-world case (JOHANNESSON; PERJONS, 2014).

Finally, according to Santos (2007), the research perspective represents the focus of interest. Therefore, this master thesis investigates the adherence of Architecture Conceptualization Process of the ISO/IEC/IEEE 42020 Standard from the point of view of architects, researchers and/or software analysts in the context of academia and software development organizations in Sergipe, state of Brazil.

Bibliographic research about software architecture conceptualization was performed to verify that some activities of this Process of ISO/IEC/IEEE 42020 were employed in an isolated way. One of the reasons for this situation is that there was no standard that disciplines the activities and tasks of an architecture conceptualization. Considering this reality, the need to use and evaluate the Conceptualization Process is justified since it is a current standard, but with a broad scope due to its list of activities and tasks.

A web application is developed to implement a framework based on the 10 activities and the 100 tasks of the Architecture Conceptualization Process. As there is a large number of activities and tasks in that Process, the goal of the tool is to receive data from the user, inputs with information of the conceptualization, i.e., *stakeholders*, *concerns* and objectives. These data are inputs for the construction of the architecture conceptualization artifacts, such as the architecture conceptualization status report, the problem space definition report and others. The tool assists in carrying out these activities and tasks in practice, regardless of the domain area.

Finally, an evaluation of the Architecture Conceptualization web application was performed. Software architects were invited to learn about this process and the documentation of the conceptualization developed was presented. A qualitative evaluation by means of a questionnaire was carried out in order to obtain information about the perceptions of these professionals regarding the Conceptualization Process, according to the Technology Acceptance Model (TAM).

Many models have been proposed to explain and predict the use of technologies such as software (VENKATESH; BALA, 2008). One of the most widely employed models for adopting and evaluating the actual use of Information Technology is the Technology Acceptance Model (TAM) (DAVIS, 1989; ADAMS; NELSON; TODD, 1992). TAM is a technology acceptance model widely used in Information Systems, which focuses on how the perception of ease of use, perception of usefulness, and social influence affect consumer attitudes.

The TAM model suggests that, when users are presented with new technology, several factors influence their decision about how and when they will use it. Two variables were first investigated in this context: *perceived usefulness*, defined as the degree to which a person believes that using an IT will enhance his or her job performance, and *perceived ease of use*, defined as the degree to which a person believes that using a particular system will be free of effort. A third variable added later, the *perceived usage* (ADAMS; NELSON; TODD, 1992), is defined as the degree to which the user will actually use the technology. A simplified representation of the TAM model is presented in Figure 15.

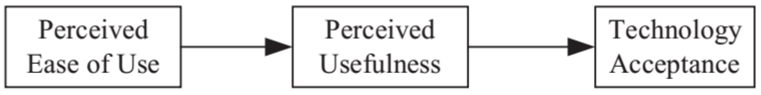


Figure 1 – Technology Acceptance Model.

1.5 Related works

As it has been recently released, there is a shortage of papers referring to ISO/IEC/IEEE 42020.

Pareto et al. (2012) describes a prioritization method, which combines collaborative and analytical techniques, and involves several stakeholder groups to produce informed recommendations, on which areas of the architecture documentation need to be improved, and what to improve in those areas. That work addresses only one of the tasks of Activity 6, "Synthesize potential solutions in the solution space", of Clause 8 (Software Architecture Conceptualization), present in the ISO/IEC/IEEE 42020 Standard (ISO/IEC/IEEE, 2019a).

Heesch, Avgeriou e Hilliard (2012) describes a framework for the documentation of architectural decisions that uses the conventions of ISO/IEC/IEEE 42010. The framework is composed of four viewpoints, related to concerns: decision details viewpoint, which mainly addresses the concerns related to the rationale that motivates decisions; viewpoint of decision's relationships, which focuses on the concerns of the relationships between decisions; viewpoint of the involvement of stakeholders in the decision, which allows explaining the relationships between stakeholders and decisions; viewpoint chronology of decisions, designed to satisfy remaining temporal concerns in the decisions. The proposed framework deals only with architectural decisions, which is one of the tasks of activity 8, “Capturing concepts and properties of architecture”, of Clause 8 (Software Architecture Conceptualization).

An architecture is presented in (KUMAR et al., 2018) for a solution that allows assistance to elderly people. The work refers to the Architecture Conceptualization of the ISO/IEC/IEEE 42020 Standard, presenting the identification activities of the stakeholders, their concerns, requirements, systems’ context, value proposition, systems’ value proposition, systems’ proposal, quality characteristics, concept maps, architectural principles, functionalities, models and visions. It is also important to note that the concept mapping of the system is not a simple task, but it can be a useful tool for the development of the system.

In (MARTIN, 2018), the author presents an overview with the key concepts of the six processes of the ISO/IEC/IEEE 42020 Standard. The author points out that the article describes the effort to incorporate software architecture practices within an international standard, serving as a support for the ISO/IEC/IEEE 15288 Standard (ISO/IEC/IEEE, 2015). The ISO/IEC/IEEE 15288 Standard presents the design definition process for each system of interest to be developed, while the ISO/IEC/IEEE 42020 Standard shows the process of defining the architecture for a complete solution given the concerns of all stakeholders. This allows the architecture to be used for solutions of various types, such as systems of systems, collection systems and product line. In (MARTIN, 2018) it is further stated that the architecture conceptualization process focuses on establishing the objectives of the architecture, and determining the fundamental concepts and properties that best meet those objectives, given the constraints, conditions and challenges.

In their turn, (PHILLIPS; MAZZUCHI; SARKANI, 2018) wrote that ISO/IEC/IEEE 42020 Standard (under development at that time) is among the documents to manage software resiliency. The Standard improves space system resiliency because it provides software architecture design, system engineering and increases software security, thereby reducing the risk of latent software defects and vulnerabilities. Resilient software architecture also contributes to higher assurance functionality.

More specifically, in (DIAMPOVESA; HUBERT; YVARS, 2021), the ISO/IEC/IEEE 42020 Standard is used in a case study of a Li-on battery for electrical vehicles, where the authors propose a design approach for architecture conceptualization. The concepts of problem space, knowledge space, and solutions space are meaningful in explaining the differences between modeling a system and modeling its design problem.

For last, Creff and others Creff et al. (2020) pointed out that Model-Based Systems Engineering (MBSE) must include assistance to the system designers in identifying candidate architectures to subsequently analyze tradeoffs, and it is expensive to build analysis models to explore the space of possible solutions. Unfortunately, efforts in the systems engineering community generally favor solution analysis, e.g. in providing system modeling languages and tools (e.g., SysML, Capella), over exploring a set of candidate architectures (architectures synthesis), a feature needed in the early stages. They then use the ISO/IEC/IEEE 42020 Standard to explore the advantages of designing and configuring the variability problem to solve one of the problems of exploring (synthesizing) candidate architectures in systems engineering.

Solutions present in (KUMAR et al., 2018; MARTIN, 2018; PHILLIPS; MAZZUCHI; SARKANI, 2018) were published considering the draft version of ISO/IEC/IEEE 42020. On the other hand, other works (CREFF et al., 2020; DIAMPOVESA; HUBERT; YVARS, 2021; QUEZADA-GAIBOR et al., 2022) presented the current version of ISO/IEC/IEEE 42020, specifically stressing out the architecture conceptualization process.

The conceptualization of the architecture aims to characterize the problem space and determine solutions that meet the concerns of the stakeholders, define the objectives of the architecture, and meet the relevant requirements. Our purpose here is to propose a framework and a web-based system to meet the needs of the architectural process present in the ISO/IEC/IEEE 42020 Standard for Architecture Conceptualization.

1.6 Structure of the work

In addition to this introductory chapter, this master thesis presents four more chapters. These chapters are described as follows:

- Chapter 2 - The ISO/IEC/IEEE 42020 Standard. It presents the main aspects of this Standard besides details about the Architecture Conceptualization Process, which is the focus of this work.
- Chapter 3 - A Proposed Framework for Architecture Conceptualization. It presents the process and results obtained from the framework.
- Chapter 4 - ArchConcept: A Web-System for Architecture Conceptualization. It presents a web system proposed to aid software architects to improve productivity and accomplish a high-quality result when following the Standard.
- Chapter 5 - Conclusions and Future Works. It presents a summary of the objectives and methods applied, contributions, limitations, future works, difficulties and conclusions about this master thesis.

2 The ISO/IEC/IEEE 42020 Standard

The ISO/IEC/IEEE 42020 Standard complements the architecture-related processes identified in ISO/IEC/IEEE 15288 (ISO/IEC/IEEE, 2015), ISO/IEC/IEEE 12207 (ISO/IEC, 2017) and ISO 15704 (ISO, 2019) with activities and tasks that enable architects and others to more effectively and efficiently implement architecture practices. Implementing these practices can help ensure that the architecture has a greater influence on business and mission success.

The Standard ISO/IEC/IEEE 42020 establishes a set of process descriptions for governance and management of a collection of architectures, as well as describes processes to architect entities. These processes are applicable both for a single project as well as for multiple projects and product lines. In addition, the Standard is useful throughout the lifespan of the architecture of software systems (ISO/IEC/IEEE, 2019a).

Architectural concepts, principles, and procedures are being applied and revised to help manage the increasing complexity faced by system stakeholders and architects about the referenced Standard. Several terms related to software architecture are defined in the standard. Those considered important are defined below:

1. *Architecting* is the process of conceiving, expressing, defining, documenting, communicating and certifying proper implementation of, maintaining and improving an architecture throughout a software's life cycle.

2. *Architecture* means fundamental concepts or properties of an entity (i.e., solution, system, subsystem, product line, family of systems, system of systems, etc.) in its environment and governing principles for the realization and evolution of this entity and its related life cycle processes.

3. *Process* is a set of interrelated or interacting activities that transforms inputs into outputs.

4. A *System* is defined as a combination of interacting elements organized to achieve one or more stated purposes.
5. A *Stakeholder* is a role, position, individual or organization having a right, share, claim or other interest in an architecture entity or its architecture that reflects their needs and expectations. The Stakeholders of a system, i.e., are parties with interests in that system, and their interests are expressed as concerns.
6. A *Concern* is a matter of interest or importance to a stakeholder. A concern could be held by one or more stakeholders. Concerns arise throughout the life cycle from system needs and requirements, design choices and from implementation and operating considerations.
7. A *Work product* is an artifact associated with the execution of a process.
8. An *Architecture description* is a work product used to express an architecture.
9. An *Architecture View* is an information item expressing the architecture from the perspective of specific stakeholders regarding specific aspects of the architecture entity and its environment. It addresses one or more of the concerns held by the system's stakeholders. An architecture view expresses the architecture of the system of interest in accordance with an architecture viewpoint (or simply, viewpoint).
10. The *Architecture viewpoint* are conventions for the construction, interpretation and use of architecture views to address specific concerns about the architecture entity (like to frame specific system concerns).
11. A *Model kind* is a convention for a type of modeling, as data flow diagrams, class diagrams, Petri nets, balance sheets, organization charts and state transition models.
12. An *Architecture rationale* records explanation, justification or reasoning about architecture decisions that have been made. The rationale for a decision can include the basis for a decision, alternatives and trade-offs considered, potential consequences of the decision and citations to sources of additional information.
13. An *Architecture framework* establishes a common practice for creating, interpreting, analyzing and using architecture descriptions within a particular domain of application or stakeholder community.

Six architecture processes are proposed in ISO/IEC/IEEE 42020, named Architecture Governance, Architecture Management, Architecture Conceptualization, Architecture Evaluation, Architecture Elaboration, and Architecture Enablement. Each one of these processes is described in terms of purpose, desired outcomes, and a list of activities and tasks for achieving those outcomes. Tasks are recommended for implementing those activities. Although the focus of this work is the Architecture Conceptualization Process, the other five processes are briefly introduced in the following sections, which were taken directly from the Standard ISO/IEC/IEEE (2019a).

2.1 Architecture Governance Process

Architecture governance and management processes are applicable to a set of architectures, typically in an enterprise or project context. During architecture governance and management, the stakeholders making financial, governance, and technical decisions are identified, documented, and included in the performance of the activities of these processes.

The purpose of the Architecture Governance process is to establish and maintain alignment of architectures in the architecture collection with enterprise goals, policies and strategies and with related architectures.

The governance process guarantees a oversight and accountability, and identifies, manages, audits and disseminates all information related to architecture collection decisions. It is applicable to all architecture efforts in an organization. Architecture governance specifies the governance directives and guidance that can be used to drive the appropriate evolution of architectures in the architecture collection. It also specifies the architecture collection goals to be pursued and establishes the strategy to be adopted for the achievement of these objectives. It uses the management plans and status regarding the architecture collection to monitor compliance with governance directives and guidance. As a result of the successful implementation of the Architecture Governance process:

- Architecture collection objectives are addressing current and anticipated business needs of the enterprise and its customers and suppliers.
- Architecting activities and decisions are aligned with enterprise and contextual concerns.
- Architecture decisions supported by rationale are made with appropriate authorities and communicated to stakeholders.
- The various architectures within the architecture collection are consistent with, and in alignment to, the goals, objectives, strategy and vision of the organization responsible for the architecting effort.
- The various architectures within the architecture collection are in alignment with each other as necessary.
- Agreement of the purpose, objectives and goals for the architecture collection is maintained amongst the key stakeholders.

2.2 Architecture Management Process

The purpose of the Architecture Management process is to implement architecture governance directives to achieve architecture collection objectives in a timely, efficient and effective manner. The management process implements governance directives and guidance and captures this in an architecture management plan. The core processes are monitored and assessed against the plan and appropriate controls are implemented to direct course corrections. Instructions and guidance are issued to the core processes as a means to provide further direction beyond what the plan provides, especially with respect to directing how the architecture(s) should be developed and used. Architecture management specifies the management plans and instructions that are used to drive the core architecture processes. It uses the execution plans and status of the core processes to monitor compliance with management plans and instructions. In addition to management instructions, management guidance is developed as a means to assist those following the plans and instructions in implementing governance directives and in carrying out work to be in better alignment with management intent.

As a result of the successful implementation of the Architecture Management process:

- Architecting activities are in alignment with architecture governance directives and guidance.
- Architecting activities are identified and prioritized, maximizing value to relevant stakeholders given available resources.
- Inputs, including stakeholder inputs, and resources for architecting are identified, made available and utilized effectively.
- Architecture collection objectives are monitored and assessed for achievement.

2.3 Architecture Enablement Process

The purpose of the Architecture Enablement process is to develop, maintain and improve the enabling capabilities, services and resources needed to perform the other architecture processes.

The Architecture Enablement process selects, modifies and develops capabilities, services and resources in support of the other processes. This process can be used to enable the development of an individual architecture or a collection of architectures, or provide enabling resources, capabilities and services to all architecture endeavors of an organization. Architecture enablement ensures that the information regarding the different enablers is uniformly organized and integrated.

As a result of the successful implementation of the Architecture Enablement process:

- Enabling capabilities, services and resources are available when and where they are needed and are accessible to those who need them.
- Enabling capabilities are suitable for and accessible to the architecture process activities.

- Enabling services are suitable for and accessible to the architecture process activities.
- Enabling resources are suitable for and accessible to the architecture process activities.
- Personnel have the requisite knowledge and skills for proper use of the enabling capabilities, services and resources.

2.4 Architecture Elaboration Process

The purpose of the Architecture Elaboration process is to describe or document an architecture in a sufficiently complete and correct manner for the intended uses of the architecture. When elaboration of the architecture is needed then the architecture objectives, architecture concepts and properties are expressed in a sufficiently complete and correct manner that can be delivered and used as the basis for more complete modeling, delineation and decomposition of these concepts and properties. Elaborated models and views, along with supporting materials, can be used to provide a more detailed understanding of the architecture and to check for consistency with the original concepts and alignment with the objectives. The elaboration process captures the architecture description in a sufficiently complete and correct manner for the intended uses and intended users of the architecture downstream from the architecture effort.

As a result of the successful implementation of the Architecture Elaboration process:

- Architecture viewpoints and metamodels are suitable for developing the appropriate architecture views and models.
- Architecture views and models are captured to an appropriate level of detail for their intended purposes, using selected architecture modeling/description languages and notations to the extent necessary for their intended use.
- Architecture views and models are accurately and completely expressed to capture the fundamental concepts and properties of the architecture to the extent necessary for their intended use.
- Architecture views and models are adequately expressed to capture the principles and precepts of the architecture that guide development and evolution of the entity being architected.
- Architecture description is under configuration control and made available to all relevant parties.
- Architecture description continues to be aligned with changes to the architecture entity during its development.

- Alignment of the architecture with relevant requirements and design characteristics is achieved.
- Alignment of the architecture with other relevant architectures is achieved.

2.5 Architecture Evaluation Process

The purpose of the Architecture Evaluation process is to determine the extent to which one or more architectures meet their objectives, address stakeholder concerns and meet relevant requirements.

During the evaluation, there can sometimes be a need for more complete models and views. In these cases, elaboration could be requested to generate additional models and views. During evaluation these models and views can be annotated with the results of the evaluation and with comments on the strengths and weaknesses of the architecture or its description.

As a result of the successful implementation of the Architecture Evaluation process:

- The basis for evaluation findings and recommendations is clearly communicated and understood by the relevant decision-makers and key stakeholders.
- Relationship between stakeholder concerns and evaluation findings and recommendations is well established.
- The projected costs, risks, opportunities and tradeoffs associated with implementing the architecture(s) are understood and well-founded.
- Stakeholders are able to understand the extent to which the architecture addresses their concerns and intended operational uses, and where and why there are shortfalls.
- Value of the architecture(s) to relevant stakeholders is understood by those doing the architecting.

2.6 Architecture Conceptualization Process

Systems are currently designed according to the IEEE 1220 Standard (IEEE, 2005). This standard specifies a design process for software-intensive systems engineering. Nevertheless, it is also used for designing physical systems. It mainly recommends the use of analysis tools for designing new products. As a result, the development of tools for synthesis has been put aside for the benefit of analysis. However, limitations have been more and more pointed out by design stakeholders, especially regarding the expression of variability. The ever-increasing complexity of new systems necessitates rethinking their design approaches, particularly during pre-design. These observations have led to the very recent ISO/IEEE 42020 Standard (ISO/IEC/IEEE, 2019a), offering a design process more focused on architecture conceptualization, i.e. the synthesis of architectures. Besides, this standard also insists on the concepts of problem and solution space. While very actual, this new design standard is not really employed as modeling and solving tools are currently lacking (DIAMPOVESA; HUBERT; YVARS, 2021).

The concepts of problem space, knowledge space and solutions space are meaningful in explaining the differences between modeling a system and modeling its design problem. That's the difference between an approach based on synthesis (in the problem space) and an approach based on analysis and simulation (in the solution space). In the problem space, designers focus on defining and representing the design problem, that relies on the specification of requirements and on the support of experts from the relevant engineering disciplines (Knowledge space). Such a representation does not attempt to describe a specific solution. For example, the problem of designing a physical system architecture is always very complex from the structural, mathematical as well as algorithmic points of view. Therefore, the most impacting decisions must be made as early as possible, ensuring that they will lead to acceptable final solutions. It is possible to control this complexity by modeling the variability and by eliminating as quickly as possible the irrelevant design choices which will not ensure the admissibility of solutions. The synthesis process thus generates one or more admissible solutions as soon as possible. (DIAMPOVESA; HUBERT; YVARS, 2021).

The purpose of the Architecture Conceptualization process is to characterize the problem space and determine suitable solutions that address stakeholder concerns, achieve architecture objectives and meet relevant requirements.

Conceptualization is the process responsible for establishing and maintaining the alignment of architectures in the architecture collection with enterprise goals, policies, strategies, and related architectures. The focus is on identifying solutions, with an emphasis on fully understanding the complete problem space. This also entails the definition and establishment of architecture objectives, as well as negotiation with key stakeholders on prioritization of their concerns (ISO/IEC/IEEE, 2019a).

Outcomes of the Architecture Conceptualization are the definition of the problem being addressed, the establishment of architectural objectives that address the key stakeholder concerns, the definition of key architecture's concepts and properties, and the principles guiding its application and evolution. Besides these outcomes, the Architecture Conceptualization process addresses key tradeoffs, as well as identifies possible limitations. Finally, within this process, the candidate's solutions are clearly defined and understood.

There are 10 activities related to the Architecture Conceptualization Process. Each activity is composed of a number of tasks that implement the activity, in a total of 100 tasks. For instance, the activity "Relate the architecture to other architectures and to relevant affected entities" is composed of 6 tasks, one of them is to "Formulate principles and precepts expected to be used during execution of the life cycle processes".

The ten activities in Architecture Conceptualization Process are:

1. Prepare for and plan the architecture conceptualization effort.
2. Monitor, assess and control the architecture conceptualization activities.
3. Characterize problem space.
4. Establish architecture objectives and critical success criteria.
5. Synthesize potential solution(s) in the solution space.
6. Characterize solutions and the tradespace.
7. Formulate candidate architecture(s).
8. Capture architecture concepts and properties.
9. Relate the architecture to other architectures and to relevant affected entities.
10. Coordinate use of conceptualized architecture by intended users.

The idea here is to deal with high-abstraction level elements related to the first activities to establish a software architecture. In this work, all 10 activities of the Architecture Conceptualization Process are considered as can be seen in Figure 2.

Work products of the Architecture Conceptualization Process are an architecture conceptualization plan, an architecture conceptualization status report, a problem space definition report, architecture objectives, a quality model, and architecture views and models.

The implementation of the Conceptualization Process produces the following outcomes.

- The problem being addressed.

  The number and availability of stakeholders vary across software projects. Considering the context of the first activities of software development, for instance, those related to Requirements Engineering - which includes requirements elicitation and prioritization -, a minimum set of stakeholders consists of users and acquirers. Complex projects can impact many users and many acquirers, each one presenting different concerns. Project requirements may necessitate including two other groups as part of the minimum set of stakeholders. First, the organization, when developing, maintaining, or operating the system or software, has a legitimate interest in benefiting from the system. Second, regulatory authorities can have statutory, industry, or other external requirements demanding careful analysis.

  User requirements are then transformed into system requirements for the system of interest. The consistent practice has shown that this process requires iterative and recursive steps in

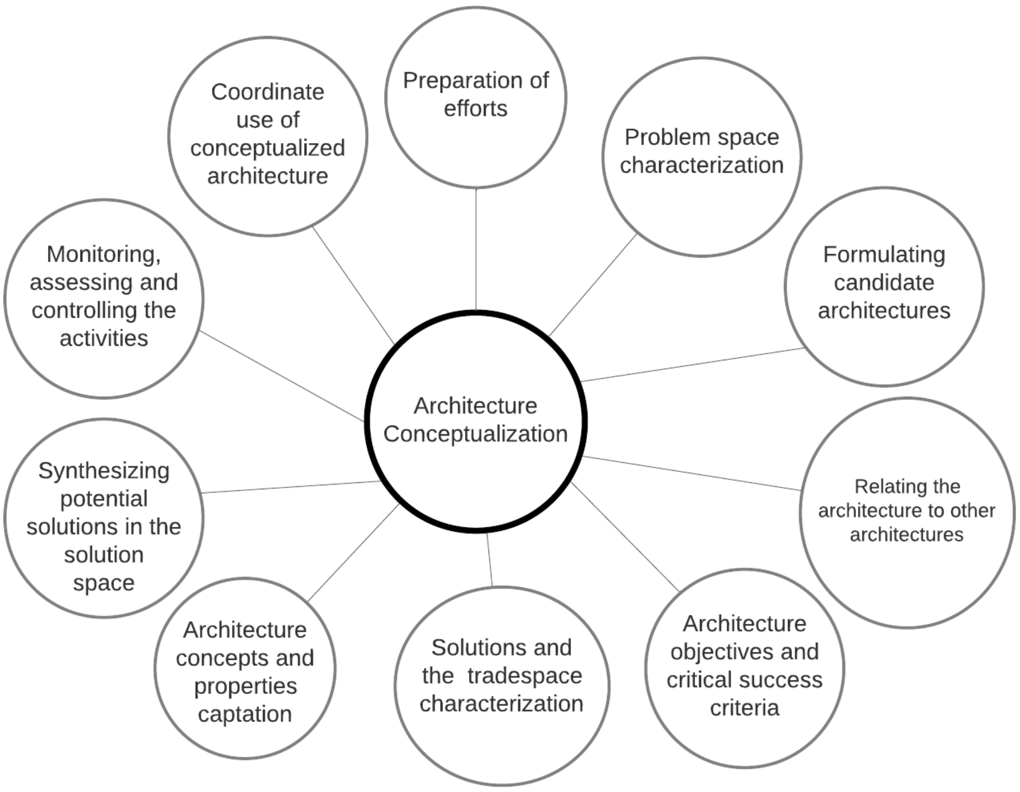

Architecture Conceptualization
Preparation of efforts
Problem space characterization
Formulating candidate architectures
Relating the architecture to other architectures
Architecture objectives and critical success criteria
Solutions and the tradespace characterization
Architecture concepts and properties captation
Synthesizing potential solutions in the solution space
Monitoring, assessing and controlling the activities
Coordinate use of conceptualized architecture

Figure 2 – Activities of the Architecture Conceptualization Process

parallel with other life cycle processes through the system design hierarchy. The recursive application of these processes will generate lower-level system element requirements (ISO/IEC/IEEE, 2017).

- Architecture objectives that address the key stakeholder concerns.

  First, activities are established to clarify stakeholders' needs and concerns. After that, it should be established the architecture objectives to meet requirements. Architecture entities are used to compose an architecture objective that deals with issues of the problem space, which can be used in other aspects of the solution. An example of an objective of the architecture of interest is to support the reuse of an entity across various technologies, protocols, platforms, operational venues, and market segments (ISO/IEC/IEEE, 2019a).

- The architecture's key concepts and properties.

  These concepts and properties can be expressed in the form of information and communication technology constructions and models such as information flows, control flows, data structures, operational rules, event/trace diagrams, state transition diagrams, timelines, and roadmaps. Also, they can be expressed in other forms such as risk models, financial models, economic models, simulation models, sensitivity models, queuing models (as well as other kinds of continuous and discrete event simulation models), geospatial models, management models, business models, social-and environmental-impact models, value stream models, among others.

- Key tradeoffs are understood concerning the problem being addressed and the relevant stakeholder concerns. Quality attributes consist of various system responses such as performance, security, availability, flexibility, and so forth. Almost every decision will affect more than one quality attribute, and thus involves a tradeoff between quality attributes, as well as the design decisions made by stakeholders in the architecture development phase, which have far-reaching consequences (OLUMOFIN; MISIC, 2007).

  Architectural design decisions that impact the quality attributes interactions are classified into sensitivity points and tradeoff points. A sensitivity point applies to a decision about specific aspects of the architecture that may affect, either benefiting or impairing at least one quality attribute; a tradeoff point is a sensitivity point between two or more quality attributes that interact in opposing ways (OLUMOFIN; MISIC, 2005).

  Stakeholders should be aware of the key tradeoffs to decide which of them will meet their concerns.

- Tradeoffs among architecture objectives and feasibility limitations are identified, and the architecture objectives targeted to be addressed by the architecture are clearly specified.

  The solution might not be addressing the entire problem or all aspects of the problem. Therefore, it is important to understand where the solutions fall short, and the tradeoffs that are considered when choosing among alternative solutions.

  If needs, wants or expectations drive the solutions, then negotiate with those stakeholders to determine which of the needs, wants, or expectations are to be translated into requirements on the solution and the relative priority of each.

- Candidate solutions for the problem are clearly defined and understood.

  In all development steps, there will be a diversity of options for solutions that address the problems. These solutions must be clear to the stakeholders so that they can decide which suits their needs.

The Activity Diagram for the Architecture Conceptualization is presented in Figure 3.

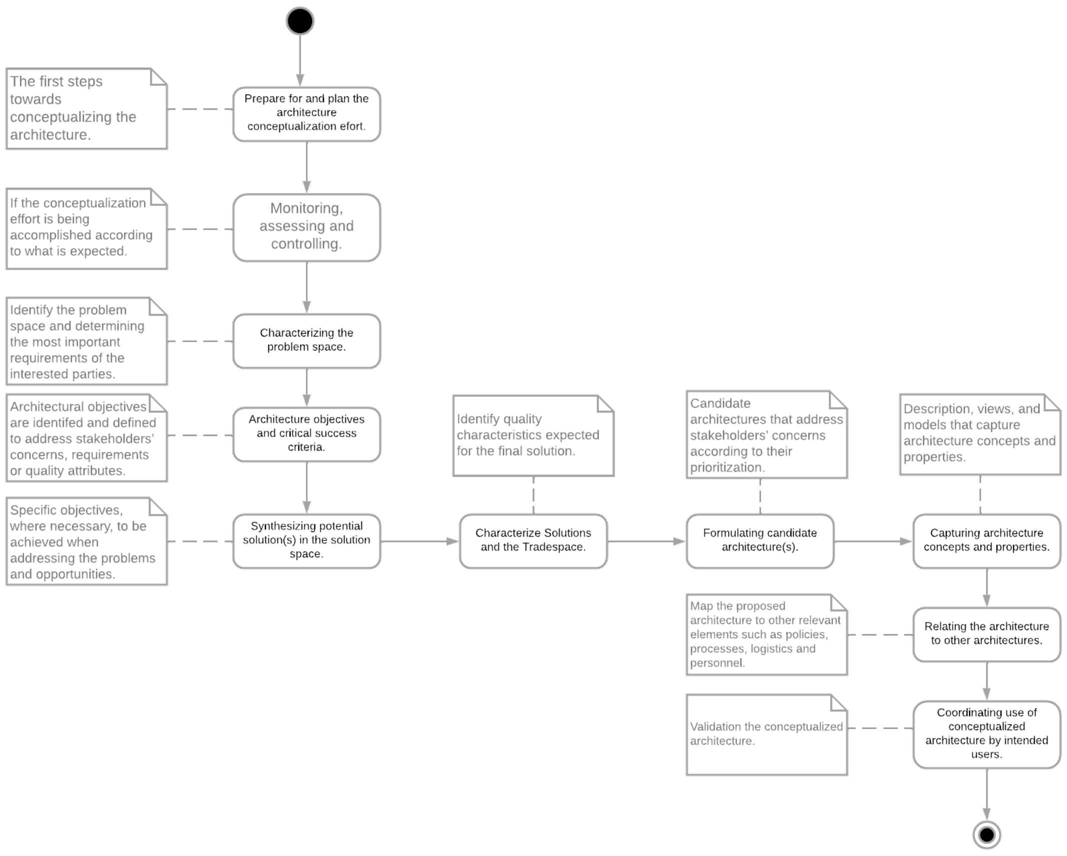

The first steps towards conceptualizing the architecture.
Prepare for and plan the architecture conceptualization efort.
If the conceptualization effort is being accomplished according to what is expected.
Monitoring, assessing and controlling.
Identify the problem space and determining the most important requirements of the interested parties.
Characterizing the problem space.
Architectural objectives are identifed and defined to address stakeholders' concerns, requirements or quality attributes.
Architecture objectives and critical success criteria.
Specific objectives, where necessary, to be achieved when addressing the problems and opportunities.
Synthesizing potential solution(s) in the solution space.
Identify quality characteristics expected for the final solution.
Characterize Solutions and the Tradespace.
Candidate architectures that address stakeholders' concerns according to their prioritization.
Formulating candidate architecture(s).
Description, views, and models that capture architecture concepts and properties.
Capturing architecture concepts and properties.
Map the proposed architecture to other relevant elements such as policies, processes, logistics and personnel.
Relating the architecture to other architectures.
Validation the conceptualized architecture.
Coordinating use of conceptualized architecture by intended users.

Figure 3 – Activity Diagram for the Architecture Conceptualization.

3 A Proposed Framework for Architecture Conceptualization

In order to create an Architecture Conceptualization, the organization should implement the relevant tasks (identified as list items under each 8.4.N activity) as appropriate to the situation, according to Clause 8 (ISO/IEC/IEEE, 2019a).

The activities adopted by the Architecture Conceptualization framework are briefly described as follows.

3.1 The Framework activities

1. Preparing for and planning the architecture conceptualization effort

   These are the first steps towards conceptualizing the architecture, which occurs through the accomplishment of some tasks. The tasks are concerned with identifying a general area of the problem, defining purpose, objectives, and level of detail, deciding which architecture description framework will be used, the architecture strategies and approaches, developing architecture conceptualization techniques, methods, and tools, and planning the architecture conceptualization effort. All these tasks need to be approved by related stakeholders. In addition, the team responsible to deal with each one of these tasks needs to be assigned.

2. Monitoring, assessing and controlling the architecture conceptualization activities

   The tasks of this activity consist of checking if the conceptualization effort is being accomplished according to what is expected, likewise checking if other processes are properly using architecture conceptualization products. This is possible through metrics, which can be checklists, key performance, risk, and opportunities indicators.

   The Standard recommends assessing and controlling the architecture conceptualization effort according to the Project Assessment and Control process present in (ISO/IEC/IEEE, 2015), which purpose is to assess if the plans are aligned and feasible; determine the status of the project, technical and process performance; and direct execution to help ensure that the performance is according to plans and schedules, within projected budgets, to satisfy technical objectives.

3. Characterizing the problem space

   Architects need to understand the basic situation that they will have to deal with for developing and managing all the software systems for which they are responsible. Therefore, some important tasks are to identify the problem space and determine the most important requirements of the interested parties. After that, the architects need to identify the solution space, which means they have to describe all the products, frameworks, patterns, services, and technologies that will address these stakeholders' problems and needs.

   It is important to adopt a systematic approach to identify the most effective solution given several possibilities. An evaluation of alternative solutions should be provided by the architecture team, and each possible solution should be documented in such a way that important architectural decisions can be retrieved whenever it is necessary. Thus, once a decision is documented, even if in the future the decision is classified as a bad one, at least it is created knowledge about the whole situation, providing an additional experience for architects, developers and other stakeholders.

4. Establishing architecture objectives and critical success criteria

   In this activity, architectural objectives are identified and defined to address stakeholders' concerns, requirements or quality attributes, which were previously identified. Critical success criteria are defined to assess whether the problem has been solved, checking whether the final objectives have been achieved. Even if the problem is not completely solved, a degree of success can be identified. It is also relevant to establish a quality model that considers the adopted quality measures as well as the relationships between them.

5. Synthesizing potential solution(s) in the solution space

   This activity proposes defining specific objectives, where necessary, to be achieved when addressing the problems and opportunities. In addition, relate these to the established architecture objectives and success criteria. Once those objectives have been part of the problem space, and are related to the architecture objectives and success criteria, it is required to assess them. On the other hand, it recommends designing a new architecture from scratch or using one previously established solution.

   It is important to store an architecture repository because the solution can be found in current or past architectures. Thus, an architecture repository also reinforces the importance of the organization of keeping an Architecture Knowledge Management. Each solution implicates characterizing strengths, weaknesses, tradeoffs, and also identifying needs, wants, and expectations for each potential solution. After that, it has to be negotiated with the stakeholders to determine which of the needs, wants or expectations are to be translated into requirements on the solution and the priority of each requirement (CAPILLA et al., 2016).

6. Characterize Solutions and the Tradespace

   When determining key criteria for system quality, it is commonly known that stakeholders will identify many quality characteristics expected for the final solution. Stakeholders will try to provide their feeling of what they need. For instance, an application needs to provide means to be secure, and also provide a minimum performance. Then, developers understand these restrictions, needs, and constraints, and establish non-functional requirements in structured documents. According to ISO/IEC/IEEE 29128 (ISO/IEC/IEEE, 2011a), non-functional requirements have to be written in such a way that they can be verified, i.e., there are means to identify if the expected quality characteristics of a system are fulfilled.

   Given the possible solutions, developers need to analyze tradeoffs between all possibilities. For each architectural relevant solution, an analysis should provide a full evaluation of the pros and cons of each solution. For instance, as soon as developers understand the necessity of using a framework, criteria of evaluation are established and the candidates are evaluated. Typically, quality characteristics need to be prioritized. For instance, for most software systems, as long as they do not need to provide strict timing constraints, performance is not as important as maintainability, or even it is better to improve the easiness of use instead of performance.

7. Formulating candidate architecture(s)

   After a solution is identified, and the functional and non-functional characteristics are described, it is time to formulate candidate architectures that address stakeholders' concerns according to their prioritization.

   It is important to create traces between key characteristics (based on identified stakeholder concerns, relevant requirements, quality attributes, architecture objectives, and other relevant factors) and the formulated candidate architectures to show how each one addresses those key characteristics. In doing that, stakeholders will have a clear understanding of each proposed architecture and be able to make better decisions.

8. Capturing architecture concepts and properties

   Architecture Conceptualization only needs to describe the architecture to the level of specificity and granularity that is suitable for its intended users, which in many cases does not require significant elaboration (ISO/IEC/IEEE, 2019a). The Elaboration Process deals with description, views, and models that capture architecture concepts and properties.

   ISO/IEC/IEEE 42020 should be used to provide more information about these architecture description concepts. These views and models are explored further in clause 10, Architecture Elaboration process, of ISO/IEC/IEEE 42020.

9. Relating the architecture to other architectures and to relevant affected entities

   Important to map the proposed architecture to other relevant elements such as policies, processes, logistics, personnel, etc. When there are relationships between architecture elements and other entities, these should be explicit. Also, these relationships must have available principles and precepts for those who will use the designed architecture. During this step, additional factors will sometimes be founded that drive the architecture, which may result in changes to the architecture. In addition, this provides an opportunity to help clarify the meaning and intent of requirements and to ensure that all relevant requirements can be met by the architecture.

10. Coordinating use of conceptualized architecture by intended users

    Key stakeholders should validate the conceptualized architecture through feedback of the description, views, and models.

    Besides, there are other users of architecture conceptualization information who are responsible for the evaluation or elaboration of the architecture, review, management, design engineering, and so on.

    Many ways can be considered for validation. For instance, workshops in which each stakeholder can give their opinions about each relevant aspect of the architecture described so far. Each item that can be improved can be identified and documented for future reference.

3.2 An Example of Architecture Conceptualization for a Health Information System

Regarding activity 1, the general nature of the problem area is a HIS capable of dealing with interoperability between legacy systems. The purpose is to define a conceptualization architecture that meets the requirements of key stakeholders and captures early decisions about high-level design. They have a special role to play in the architecture development effort because they will deal with the resulting solution for years to come.

Regarding activity 2, a project planning timeline, a Kanban board or a Scrum board can be used to monitor, evaluate and control the architecture conceptualization activities of the HIS.

Regarding activity 3, concerning to characterization of the problem space, patients may need reminders or help to take their medications, and often need assistance with ambulation (walking) and transferring from a bed to a chair or wheelchair, or getting in and out of the shower. Many patients have adaptive equipment, such as walkers, wheelchairs, canes, and prosthetic devices, which assist them in moving around their home (MCLAIN; O'HARA-LESLIE; WADE, 2016). IRELAND

Existing HIS are sometimes proprietary, monolithic, high coupling, and expensive solutions for patients and their relatives. Most of such systems do not consider their inter-operation with existing, distributed, and external systems, such as Electronic Health Records(EHR), Patient Health Records (PHR), emergency systems (e.g., ambulances, fire departments), and other HIS. For the healthcare domain, there are guidelines to construct HIS that provide support to chronic disease management of patients at home. As HIS deal with human life, they have to be designed considering important non-functional requirements including interoperability, security, safety, performance and reliability.

Patients are key stakeholders of HIS. An example of key patient user requirements (high level of abstraction) is as follows:

- Patients shall have all the relevant information needed to allow the management of their own health and their interaction with the health system.
- Patients shall be recognized when they access the system and can quickly see relevant health details.
- Patients shall access their own health records and maintain a health diary.
- Patients shall have high levels of trust in the security and confidentiality of the services they are using.
- Patients shall have the ability to access services while traveling and on the move.

Care Providers are also key stakeholders. An example of key user requirements (high level of abstraction) is as follows:

- Care providers shall have the ability to constantly monitor and interact with patients despite the distance and mobility of either party.
- The healthcare delivery environment shall be safe and supportive of patient care.
- Care providers shall access information from other systems to support their decision.
- Care providers shall be able to interact with patients regardless of their electronic devices.

Considering these two important stakeholders, related concerns for each one are as follows:

- Patients: Affordability, Availability, Dependability, Reliability and Usability.
- Care providers: Availability, Flexibility, Maintainability, Reliability, Usability, Resilience.

According to activity 4, some architecture objectives and architectural significant requirements are described as follows.

- Domain requirements. The reference architecture must enable the development of HIS that, remotely, estimates and provide the patient's physical status at any time, e.g., informing the status of the patient through the analysis of his/her physiological functions and body systems (e.g., cardiovascular, nervous, respiratory) signs and symptoms.
- Interoperability and integration requirements. The architecture must enable the development of HIS that allows interoperable communication between legacy systems.
- Reliability requirements. The reference architecture must enable the development of HIS that offers fault-tolerant mechanisms for constituent systems interactions.

Another work product is a quality model. SOA Quality Model (SOAQM) (FRANCA; SOARES, 2015; FRANCA; LIMA; SOARES, 2017) will be adopted. This model is based on ISO/IEC/IEEE 25010 and has quality attributes that are considered relevant during the development of SOA applications. According to HIS objectives, the following characteristics of SOAQM will be measured:

- Compatibility.
- Usability.
- Reliability.
- Security.

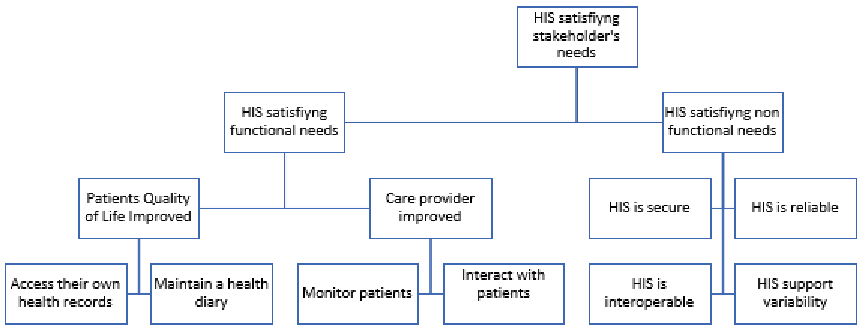

HIS satisfiyng stakeholder's needs
HIS satisfiyng functional needs
HIS satisfiyng non functional needs
Patients Quality of Life Improved
Care provider improved
HIS is secure
HIS is reliable
Access their own health records
Maintain a health diary
Monitor patients
Interact with patients
HIS is interoperable
HIS support variability

Figure 4 – High level mission view of HIS.

Regarding activity 5, a new architecture conceptualization is being proposed for a HIS. Although there are other HIS, they aren't available for use. The Architecture Knowledge is stored manually because there is no proper software available.

Regarding activity 6, the one solution proposed is a HIS whose stakeholders concerns in common are: Availability, Reliability and Usability. It means they should be prioritized.

Regarding activity 7, if would be many variables to be considered, it would generate candidate architectures and should be formulated. For example, a candidate architecture focused on the Patient's concerns instead of the Care provider's ones.

Concerning activity 8 and the architecture views and models as a work product, Figure 4 describes a high-level mission view of HIS.

Regarding activity 9, there was not found a HIS architecture according to the Architecture Conceptualization Process.

Regarding activity 10, forms are used to register feedback from users.

4 ArchConcept: A Web-System for Architecture Conceptualization

An architecture is typically developed because key people have concerns that need to be addressed by the business and IT systems within the organization. Such people are commonly referred to as the “stakeholders” for the system. One important role of the architect is to address these concerns, by identifying and refining the requirements that the stakeholders have, developing views of the architecture that show how the concerns and the requirements are going to be addressed, and showing the tradeoffs that are going to be made in reconciling the potentially conflicting concerns of different stakeholders. Without the architecture, it is unlikely that all the concerns and requirements will be considered and met.

As described in the previous Chapter, the Conceptualization Architecture Process of the ISO/IEC/IEEE 42020 Standard presents many high-level abstract definitions. Due to the high amount of tasks to be performed about Architecture Conceptualization based on this Standard, the software architect has to spend time to understand what is proposed in the Standard, get stakeholders’ information, organize and gather all knowledge obtained concerning Clause 8.

Considering these needs, a web system is proposed to aid software architects to improve productivity and accomplish a high-quality result by following the Standard’s recommendation through the use of this tool. This Chapter describes the Architecture Description according to the ISO/IEC/IEEE 42010 Standard, the Architecture Conceptualization according to ISO/IEC/IEEE 42020 Standard and the design of the ArchConcept, and, at last, the evaluation of the ArchConcept using TAM.

4.1 ArchConcept Requirements

The functional requirements were obtained from the main activities of the framework, which leads to capturing stakeholders’ (in this study, the architect) needs, including their main concerns and related viewpoints, problem space, architecture objectives, architecture decisions, and tradeoffs.

- RF01 - The system should allow the software architect to register the problem.
- RF02 - The system should allow the software architect to change the problem.
- RF03 - The system should allow the software architect to delete the problem.
- RF04 - The system should allow the software architect to register the stakeholders.
- RF05 - The system should allow the software architect to change stakeholders.
- RF06 - The system should allow the software architect to exclude stakeholders.
- RF07 - The system should allow the software architect to register the concerns.
- RF08 - The system should allow the software architect to change the concerns.
- RF09 - The system should allow the software architect to delete the concerns.
- RF10 - The system should allow the software architect to register the objectives of the architecture.
- RF11 - The system should allow the software architect to change the objectives of the architecture.
- RF12 - The system should allow the software architect to delete the objectives of the architecture.
- RF13 - The system should allow the software architect to register viewpoints.
- RF14 - The system should allow the software architect to change viewpoints.
- RF15 - The system should allow the software architect to delete viewpoints.
- RF16 - The system should allow the software architect to register the tradeoffs.
- RF17 - The system should allow the software architect to change tradeoffs.
- RF18 - The system should allow the software architect to delete tradeoffs.
- RF19 - The system should allow the software architect to register architectural decisions.
- RF20 - The system should allow the software architect to change architectural decisions.
- RF21 - The system should allow the software architect to delete architectural decisions.

4.2 ArchConcept Architecture Description according to the ISO/IEC/IEEE 42010 Standard

The ISO/IEC/IEEE 42010 Standard ISO/IEC/IEEE (2011b) does not specify any format for recording architecture descriptions, but it describes the basic context of an architecture description. Besides, the Standard specifies the best practices for documenting enterprise, system and software architectures.

Considering the ArchConcept architecture description according to the ISO/IEC/IEEE Standard ISO/IEC/IEEE (2011b) presented in this work, the software architects are the stakeholders and have interests in a software tool (one or more software systems). These interests are the concerns, such as Functionality and Usage. A concern pertains to any influence on a system in its environment. Each system is situated in an environment, which is a context determining the setting and circumstances of all influences upon a software system. The ArchConcept is in the operational environment.

Software architecture is documented in multiple views in order to help stakeholders to better understand the software system. In addition, architecture description using multiple views allows for managing complexity and risks during software development. Four architectural views, Scenario, Business Process, Logical and Development, are presented to provide an overview of the development of the application. Each view presents different elements, improving the separation of concerns.

4.2.1 ArchConcept Architectural Decisions

Architectural Description is also a set of Architectural Decisions, that guide high-level project development. Among them, Tables 1, 2 and 3 present examples of Architectural Decisions for the ArchConcept application. Architectural Decision for ArchConcept application D01 is concerned with providing more options of accessibility to use ArchConcept. Architectural Decision for ArchConcept application D02 deals with information security. Architectural Decision for ArchConcept application D03 is concerned with using Open Source tools to develop the system.

Table 1 – Architectural Decision for ArchConcept application D01.

Description	Accessibility decision.
Constraint	Minimum requirements .
Solution	The development view of architecture should be in layers, provided by the MVC pattern
Rationale	Using the system from different devices will provide more flexibility and increase the possibility of access.

Table 2 – Architectural Decision for ArchConcept application D02.

Description	Security decision.
Constraint	Don’t overload the system.
Solution	Security must be addressed to protect information from unauthorized people and avoid losing information.
Rationale	Defining strategies for protecting the access of unauthorized persons and for avoiding loss of information.

Table 3 – Architectural Decision for ArchConcept application D03.

Description	Platafom, language, and tool preferences.
Constraint	Only open source tools should be used for development.
Solution	Define open source tools to development.
Rationale	The use of open-source development tools due to high quality and mature use and due to an experimental web application.

4.2.2 ArchConcept Architectural Views

The architectural views Scenario, Business Process, Logical and Development were chosen because they provide the essential views so that the application can be comprehended at a high level and communicated for design and development activities.

4.2.2.1 Scenarios View

The Use Case diagram of the proposed system provides a high-level view, gathering the scenarios of execution, as depicted in Figure 5. The software architect is able to manage the functionalities of the system, which involves managing: Viewpoint, Concern, Stakeholder, Problem, Objective, Decision and Tradeoffs. These functionalities are related to the purpose of the Architecture Conceptualization process, which is to characterize the problem space and determine suitable solutions that address stakeholder concerns and are related to the main outcomes of the Architecture Conceptualization: the definition of the problem being addressed, the establishment of architectural objectives that address the key stakeholder concerns and the definition of key architecture concepts and properties, which guide its application and evolution.

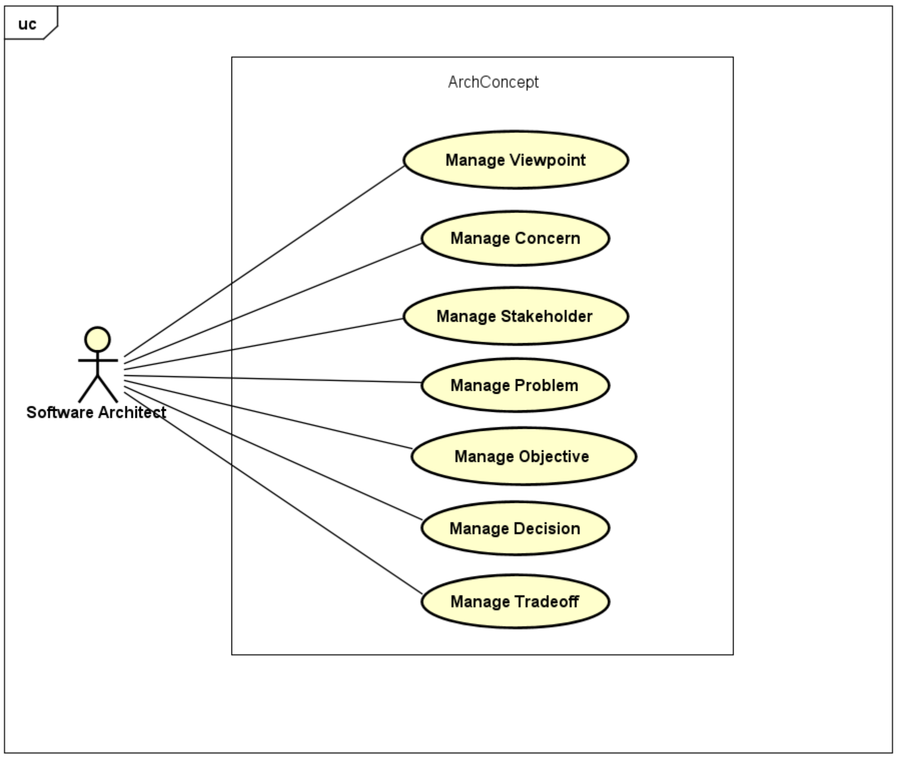


Figure 5 – UML Use Case diagram of the system for architecture conceptualization.

4.2.2.2 Business Process View

The UML Activity diagram in Figure 6 presents part of the Architecture Conceptualization process implemented by the ArchConcept. As an experimental system, the only process presented is the proceedings of the architect to fill in some information about the Architecture Conceptualization, partially modeled and developed in the ArchConcept.

In Figure 6, first, the architect registers the concern, then the stakeholder related to that concern. Following, he/she registers the viewpoint that frames the concern. The architect registers the problems, which are expressed in architectural objectives. The tradeoffs registered characterize the architectural decisions guided through the architectural objectives.

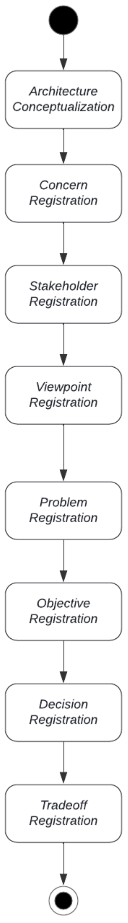


Figure 6 – UML Activity diagram - Architecture Conceptualization process.

4.2.2.3 Development View

Development of the ArchConcept application is based on the Model View Controller (MVC) pattern, as depicted in Fig 7. ArchConcept source code is organized into three modules that are model, view and controller. The model considers components that directly manage the data, logic and rules of the application using Spring Framework and Java Persistence API (JPA). View generates output representation of ArchConcept system using two frameworks that are Thymeleaf and Materialize. Control addresses input manipulation and converts it to commands for the model or view. Control represents components in ArchConcept responsible for flow control and business rules.

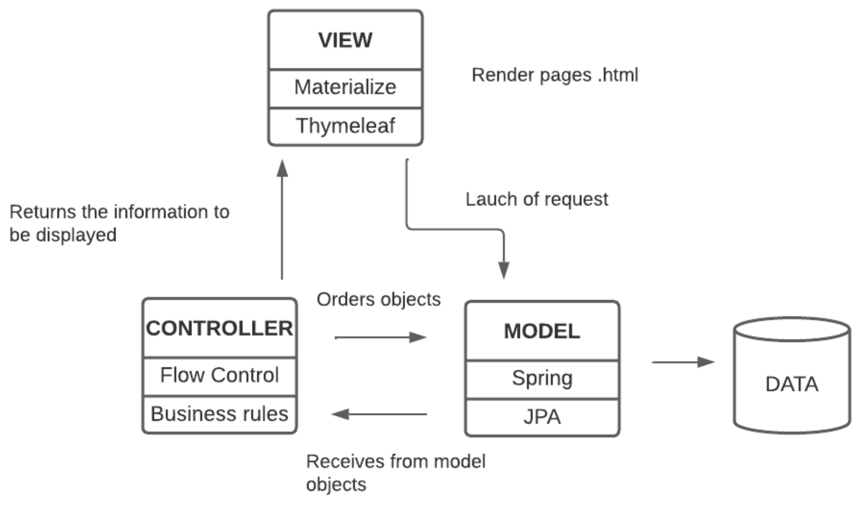


Figure 7 – MVC model for application implementation.

The Java Spring Boot MVC, a MVC-Based Framework, is used to develop the system, along with Open Source technologies, such as Thymeleaf and JPA, for the persistence of data in the PostgreeSQL database.

ArchConcept includes the use of free tools and technology as presented in Table 4.

Table 4 – ArchConcept tools and technology.

Technology	**Layer**	**Role**
PostgreSQL	Data	Relational Database
Hibernate	Distribution	Object-Relational Mapping
Thymeleaf, materialize	Front End	User interface communication
Apache Maven	Deploy	IDE confguration for Deploy
Spring Tools 4 for Eclipse	Coding	IDE

The application was deployed to Heroku, a container-based cloud Platform as a Service (PaaS). The app is available at https://arch-concept.herokuapp.com/login URL.

4.2.2.4 Logical View

The UML Class diagram is applied as a basis for developing the ArchConcept system. This class diagram was used to develop Java classes of the ArchConcept application. A stakeholder has an interest in concerns, which are framed by viewpoints. Also, a stakeholder is related to the identification of problems, which is expressed in architectural objectives. Tradeoffs characterize the architectural decisions guided through the architectural objectives. The Class diagram in Figure 8 presents the classes Viewpoint, Concern, Stakeholder, Problem, Objective, Decision and Tradeoff, their attributes and the relationships among them.

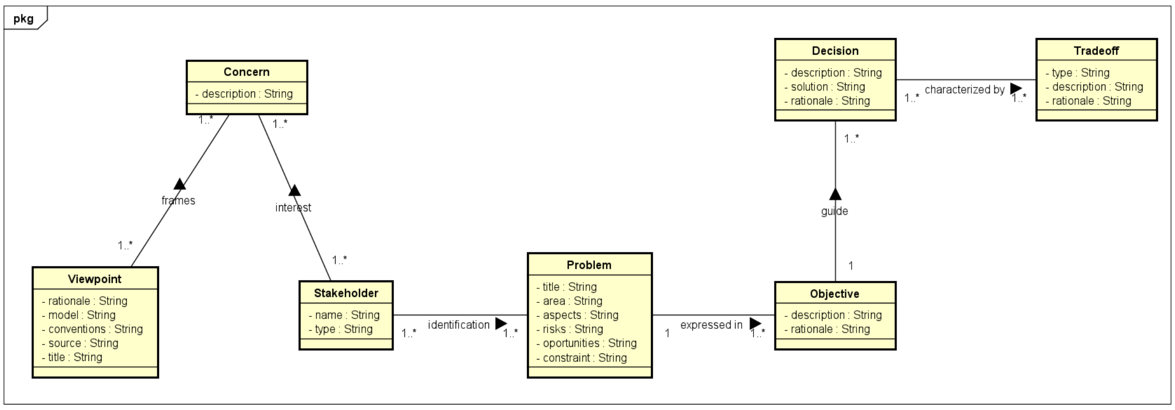


Figure 8 – Class Diagram of the system for architecture conceptualization.

4.2.3 ArchConcept Viewpoint

Architectural Description is also a set of Viewpoints, that describe the system from different perspectives. Just to exemplify a Viewpoint, Table 5 presents the operation viewpoint for the ArchConcept application.

Table 5 – ArchConcept Operation Viewpoint.

Name	ArchConcept Operation Viewpoint
Overview	The architecture viewpoint deals with the main stakeholder concerns related to the ArchConcept operation.
Stakeholders	Architect(s) who design and describe the architecture
Concern	Functionality Usage
Environment	Operational

4.3 ArchConcept Functionalities

Sample screens are presented describing the main functionalities of the ArchConcept and examples of inputs.

1. Concern registration

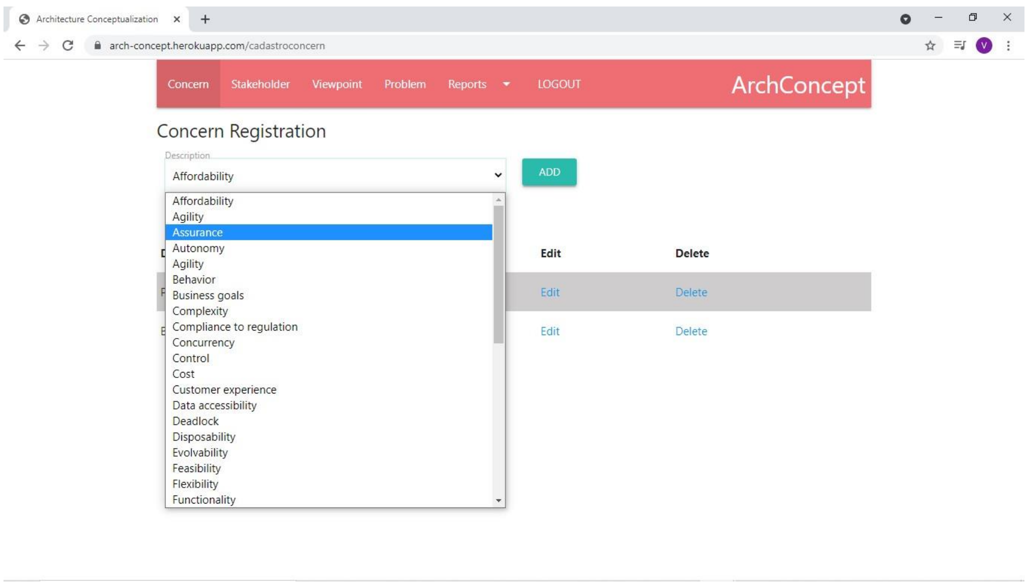
   Figure 9 presents the screen with the concern registration form. It is possible to register, edit or delete the concerns. The concerns considered are in accordance with the ISO/IEC/IEEE 42010 Standard.

   - Input Example:

   Concern: Functionality, Usage.

2. Stakeholder registration

   Figure 10 presents the screen with the stakeholder registration form. It is possible to register, edit or delete the stakeholders.


Architecture Conceptualization
arch-concept.herokuapp.com/cadastroconcern
Concern
Stakeholder
Viewpoint
Problem
Reports
LOGOUT
ArchConcept
ADD

Concern Registration

Description

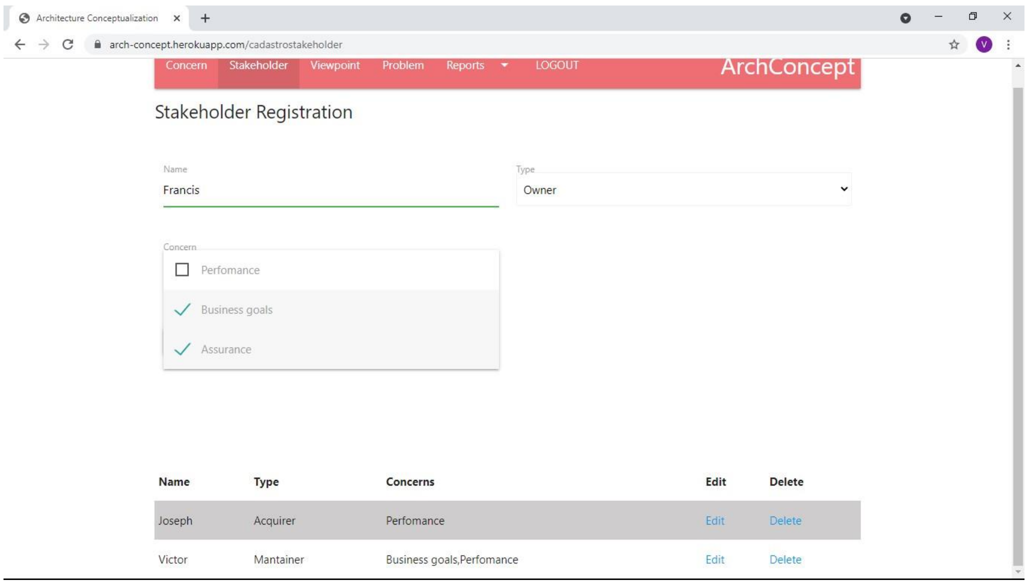
Affordability

Affordability
Agility
Assurance
Autonomy
Agility
Behavior
Business goals
Complexity
Compliance to regulation
Concurrency
Control
Cost
Customer experience
Data accessibility
Deadlock
Disposability
Evolvability
Feasibility
Flexibility
Functionality

Edit	Delete
Edit	Delete
Edit	Delete

Figure 9 – Concern Registration screen.

The type and the concern are previously registered. The types are provided, according to the ISO/IEC/IEEE 42010 Standard.


Architecture Conceptualization
arch-concept.herokuapp.com/cadastrostakeholder
Concern
Stakeholder
Viewpoint
Problem
Reports
LOGOUT
ArchConcept

Stakeholder Registration

Name: Francis

Type: Owner

Concern

☐ Perfomance
✓ Business goals
✓ Assurance

Name	Type	Concerns	Edit	Delete
Joseph	Acquirer	Perfomance	Edit	Delete
Victor	Mantainer	Business goals,Perfomance	Edit	Delete

Figure 10 – Stakeholder Registration screen.

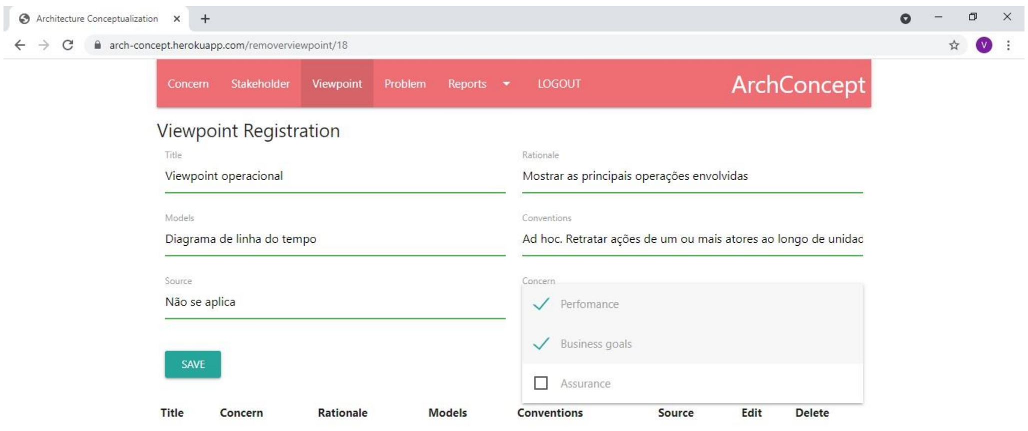
- Input Example:

Stakeholder: Software Architect. Type: User.

3. Viewpoint registration

   Figure 11 presents the screen with the viewpoint registration form. It is possible to register, edit or delete the viewpoint.

   The Viewpoint's title, rationale, models, conventions and source are provided, besides it needs to select a Concern (previously registered in the Concern's screen). The registered data will be displayed in a table at the bottom of the screen.


Architecture Conceptualization
arch-concept.herokuapp.com/removerviewpoint/18
Concern
Stakeholder
Viewpoint
Problem
Reports
LOGOUT
ArchConcept
SAVE

Viewpoint Registration

Title: Viewpoint operacional

Rationale: Mostrar as principais operações envolvidas

Models: Diagrama de linha do tempo

Conventions: Ad hoc. Retratar ações de um ou mais atores ao longo de unidac

Source: Não se aplica

Concern: ✓ Perfomance ✓ Business goals ☐ Assurance

Title	Concern	Rationale	Models	Conventions	Source	Edit	Delete

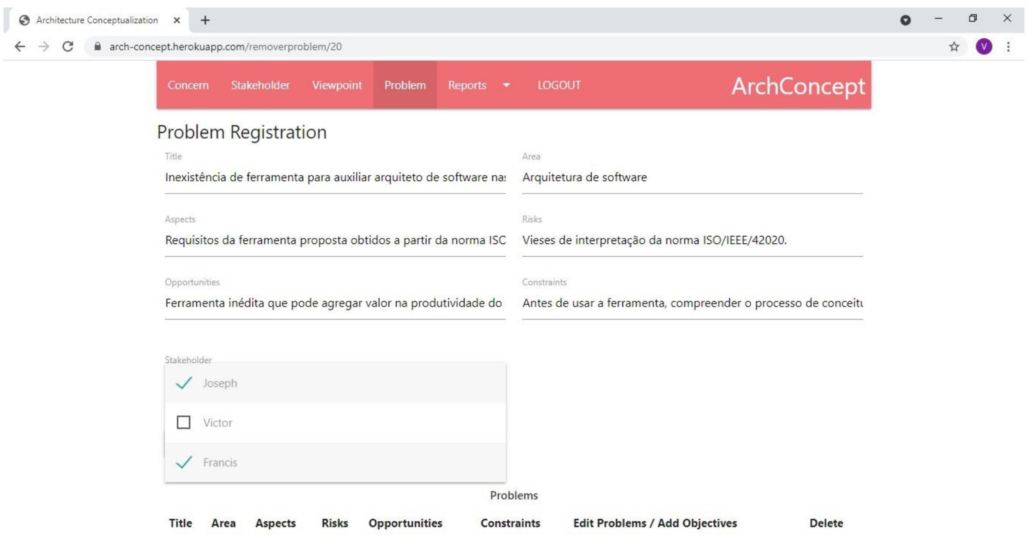
Figure 11 – Viewpoint Registration screen.

- Input Example:

Title: Operational viewpoint. Rationale: Show the main operations involved. Model: Timeline diagram. Conventions: Ad hoc. Portray actions of one or more actors over units of time. Source: not applicable. Concern: Functionality, Usage.

4. Problem registration

   Figure 12 presents the screen for the problem registration form. It is possible to register, edit or delete the problem.


Architecture Conceptualization
arch-concept.herokuapp.com/removerproblem/20
Concern
Stakeholder
Viewpoint
Problem
Reports
LOGOUT
ArchConcept

Problem Registration

Title: Inexistência de ferramenta para auxiliar arquiteto de software na:	Area: Arquitetura de software
Aspects: Requisitos da ferramenta proposta obtidos a partir da norma ISC	Risks: Vieses de interpretação da norma ISO/IEEE/42020.
Opportunities: Ferramenta inédita que pode agregar valor na produtividade do	Constraints: Antes de usar a ferramenta, compreender o processo de conceitu

Stakeholder
- ✓ Joseph
- ☐ Victor
- ✓ Francis

Problems

Title	Area	Aspects	Risks	Opportunities	Constraints	Edit Problems / Add Objectives	Delete

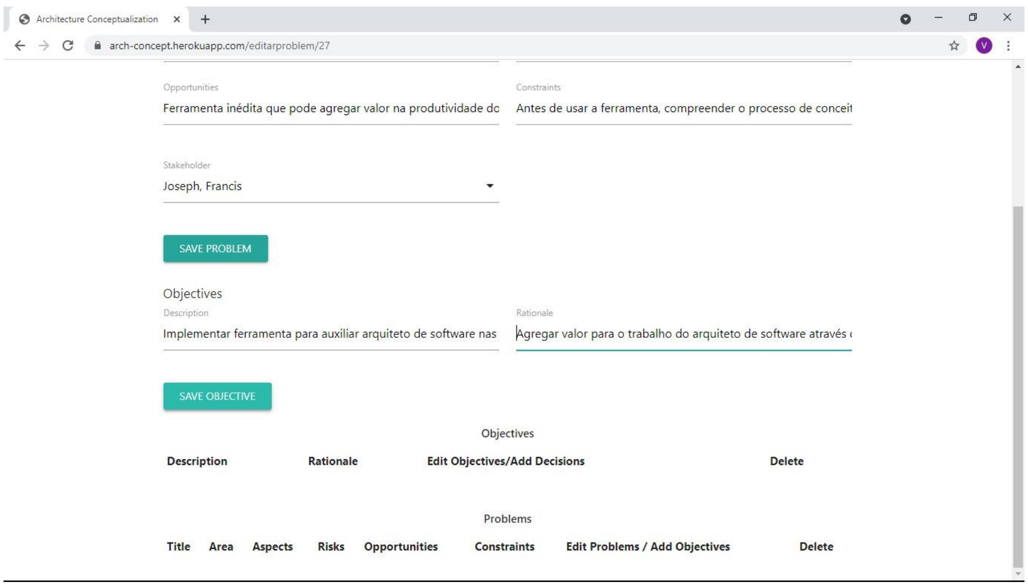
Figure 12 – Problem Registration screen.

- Input Example:

Title: Nonexistence of a system to help software architects in the activities and tasks of conceptualization of the software architecture Area: Software Architecture. Aspects: Requirements of the proposed system obtained from the ISO/IEEE/42020 Standard. Risks: Biases in the interpretation of the ISO/IEEE/42020 Standard. Opportunities: New tool that can add value to the software architect's productivity concerning the architecture conceptualization process of the ISO/IEEE/42020. Constraints: Before using the tool, understand the conceptualization process of the the architecture of the ISO/IEEE/42020 Standard. Stakeholder: Software Architect.

5. Objective registration

   Figure 13 presents the screen for the objective registration form. It is possible to register, edit or delete the objective.


Architecture Conceptualization
arch-concept.herokuapp.com/editarproblem/27
SAVE PROBLEM
SAVE OBJECTIVE

Opportunities: Ferramenta inédita que pode agregar valor na produtividade dc

Constraints: Antes de usar a ferramenta, compreender o processo de conceit

Stakeholder: Joseph, Francis

Objectives

Description: Implementar ferramenta para auxiliar arquiteto de software nas

Rationale: Agregar valor para o trabalho do arquiteto de software através

Objectives

Description	Rationale	Edit Objectives/Add Decisions	Delete

Problems

Title	Area	Aspects	Risks	Opportunities	Constraints	Edit Problems / Add Objectives	Delete

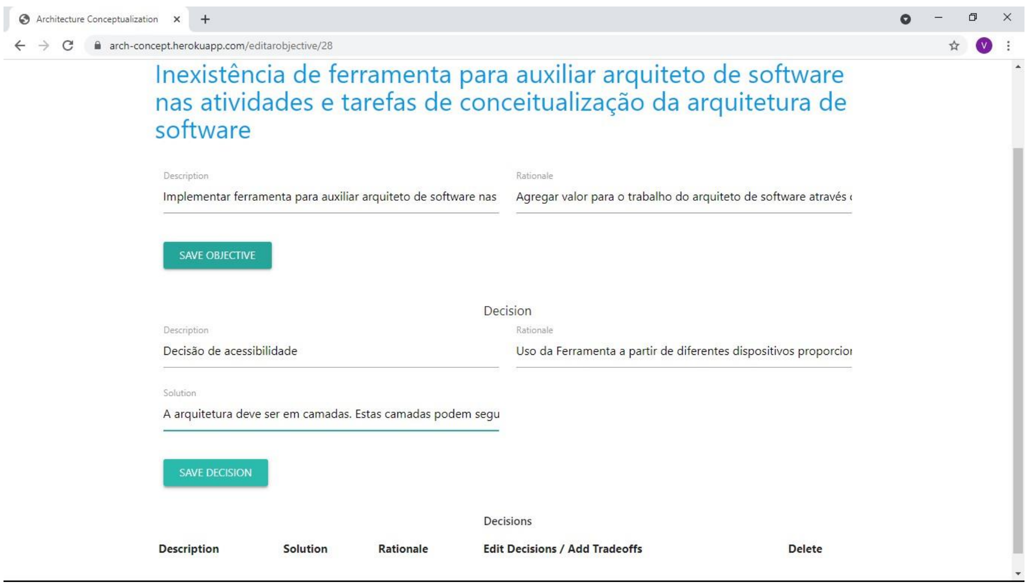
Figure 13 – Objective Registration screen.

- Input Example:

Description: Develop a system to help software architects in the activities and tasks of software architecture conceptualization. Rationale: To add value to the software. architect's work through the facility for generation and maintenance of the software architecture documentation, regarding the conceptualization of the software architecture.

6. Decision registration

   Figure 14 presents the screen for the decision registration form. It is possible to register, edit or delete the decision.


Architecture Conceptualization

arch-concept.herokuapp.com/editarobjective/28

Inexistência de ferramenta para auxiliar arquiteto de software nas atividades e tarefas de conceitualização da arquitetura de software

Description: Implementar ferramenta para auxiliar arquiteto de software nas	Rationale: Agregar valor para o trabalho do arquiteto de software através (

SAVE OBJECTIVE

Decision

Description: Decisão de acessibilidade	Rationale: Uso da Ferramenta a partir de diferentes dispositivos proporcioi
Solution: A arquitetura deve ser em camadas. Estas camadas podem segu	

SAVE DECISION

Decisions

Description	Solution	Rationale	Edit Decisions / Add Tradeoffs	Delete

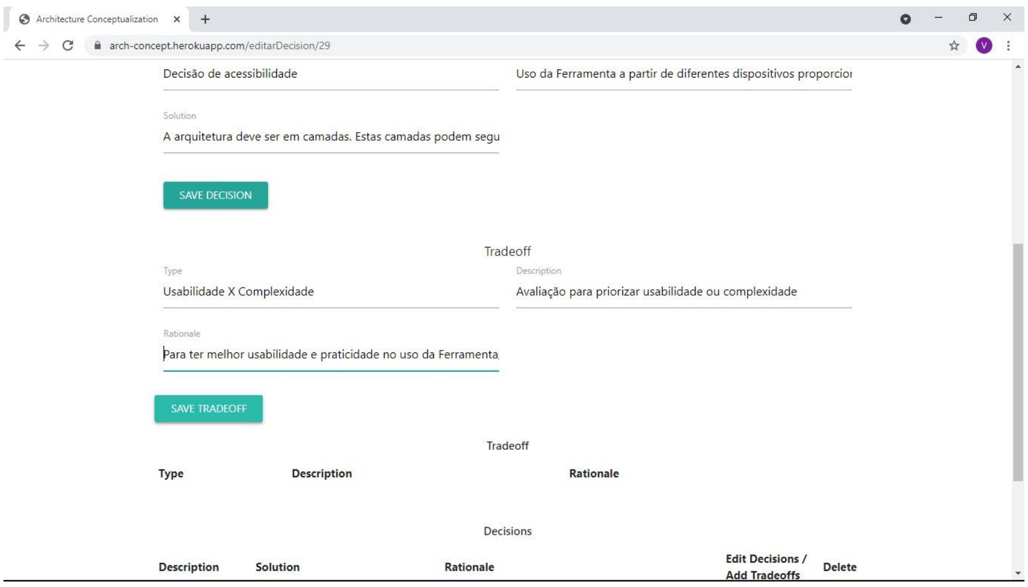
Figure 14 – Decision Registration screen.

- Input Example:

Description: Accessibility decision Solution: The development view of architecture should be in layers, provided by the MVC pattern. Rationale: Using the system from different devices will provide more flexibility and increase the possibility of access.

7. Tradeoff registration

   Figure 15 presents the screen for the tradeoff registration form. It is possible to register, edit or delete the tradeoff.


Architecture Conceptualization
arch-concept.herokuapp.com/editarDecision/29

Decisão de acessibilidade

Uso da Ferramenta a partir de diferentes dispositivos proporcio

Solution

A arquitetura deve ser em camadas. Estas camadas podem segu

SAVE DECISION

Tradeoff

Type	Description
Usabilidade X Complexidade	Avaliação para priorizar usabilidade ou complexidade

Rationale

Para ter melhor usabilidade e praticidade no uso da Ferramenta,

SAVE TRADEOFF

Tradeoff

Type	Description	Rationale

Decisions

Description	Solution	Rationale	Edit Decisions / Add Tradeoffs	Delete

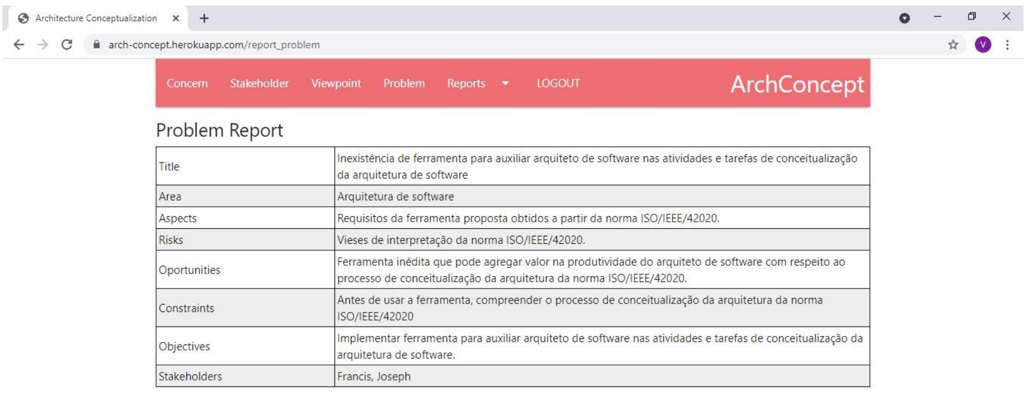
Figure 15 – Tradeoff Registration screen.

- Input Example:

Type: Usability vs. Complexity. Description: Evaluation to prioritize usability or complexity. Rationale: Provide better usability and practicality when using the Tool, as it is hardly possible to check in detail all the 100 tasks of the architecture conceptualization process.

8. Reports

Figure 16 presents the screen for the reports. It is possible to visualize all the registered information on one screen.


Architecture Conceptualization
arch-concept.herokuapp.com/report_problem
Concern
Stakeholder
Viewpoint
Problem
Reports
LOGOUT
ArchConcept

Problem Report

Title	Inexistência de ferramenta para auxiliar arquiteto de software nas atividades e tarefas de conceitualização da arquitetura de software
Area	Arquitetura de software
Aspects	Requisitos da ferramenta proposta obtidos a partir da norma ISO/IEEE/42020.
Risks	Vieses de interpretação da norma ISO/IEEE/42020.
Oportunities	Ferramenta inédita que pode agregar valor na produtividade do arquiteto de software com respeito ao processo de conceitualização da arquitetura da norma ISO/IEEE/42020.
Constraints	Antes de usar a ferramenta, compreender o processo de conceitualização da arquitetura da norma ISO/IEEE/42020
Objectives	Implementar ferramenta para auxiliar arquiteto de software nas atividades e tarefas de conceitualização da arquitetura de software.
Stakeholders	Francis, Joseph

Figure 16 – Problems Report screen.

4.4 Evaluation of the ArchConcept using TAM

In this research, we are interested in the individual's acceptance and further use of a new web-based application for architecture conceptualization, which is a good starting point to view how this new technology might be accepted in a community. The evaluation was performed using a technique survey, which was based on a questionnaire using the TAM theory with practitioners and researchers.

As explained previously, Technology Acceptance Model (TAM) is a model of IT adoption and use. It is adopted in this work for part of the evaluation.

4.4.1 The Questionnaire

The participants, researchers or practitioners, were invited according to their professional area related to Software Engineering and Software Architecture. They work as Researchers, System Analysts, University Professors, and Project Managers, and on average have 9 years of experience in Software Developing and more than 4 years of experience in Software Architecture.

After reading the user guide (also available in the appendix of this work) that is present on the initial page of the ArchConcept system, the participants used Google Forms to answer a questionnaire composed of 33 statements (11 of Perceived usefulness, 17 of Perceived ease of use, and 5 Perceived of usage of the ArchConcept system) in which they made their opinions explicit.

A 5-point Likert scale (LIKERT, 1932) was proposed to measure the perceived attitudes of the employees by providing a range of responses to each statement. The scale ranged from (1) strongly disagree, (2) disagree, (3) neutral, (4) agree, and (5) strongly agree. For the negative statements, it was inferred from low scores that the participants are positive about a statement in the questionnaire, while for the positive statements, a high score was inferred that the participants were positive about the statement.

4.4.2 Questionnaire results

The questionnaire was responded by seventeen personnel, working in the industry and as researchers/university professors. The answer regarding each question is shown in Tables 6, 7, and 8, in which "p" indicates the number of positive answers. We arbitrarily considered as positive the answers "Agree" or "Strongly Agree" (values 4 or 5). For negative sentences, we considered 1 and 2 as positive responses.

Table 6 presents answers related to statements 1 to 11 and indicates that ArchConcept presents useful functionalities, in which 72,7 % of the statements have positive answers. Statements 1, 4, 5, 8, and 9 are top-rated with 12 positive answers. 12 respondents answered that using ArchConcept allows them to better manage their work activities, and the same number of respondents would use ArchConcept functionalities like managing concerns, viewpoints, and decisions. 11 subjects answered ArchConcept is useful to manage architecture problems and objectives. 10 subjects marked that stakeholders' functionalities in the ArchConcept are useful for their work. On the other hand, only 8 answered that ArchConcept increases their performance and productivity, and that enables them to perform their tasks in less time.

Table 7, concerning answers related to statements 1 to 17, indicates that ArchConcept is easy to use. 100 % of the statements were positive answers. Statements 1, 2, 3, 4, 5, 6, and 13 are top-rated with 16 positive answers. For 16 respondents, it is easy to manage stakeholders, viewpoints, problems, correlate stakeholders and concerns, register a viewpoint from the concern already registered, set up a problem, and associate it with one or more stakeholders. For 15 participants, it is easy for them to remember how to execute tasks in ArchConcept and use it. For 14 participants, it is easy to manage objectives. For 13 participants, it is easy to manage decisions and to add one or more decisions from a previously registered objective. For 12 participants, it is easy to manage tradeoffs, register one or more objectives from an already registered problem, and register one or more tradeoffs from a previously registered decision. And for 11 participants, using ArchConcept made their work easier.

Table 8, concerning the answers related to statements 1 to 5, assesses the perceived usage of the ArchConcept. 80% of the statements have positive answers. 13 respondents were interested in knowing more about the Architecture Conceptualization process, present in ISO/IEC/IEEE 42020. 12 would use at least one of ArchConcept's functionalities in their work. For 10 subjects, the reports generated by ArchConcept are useful, and they would use most or all of the functionalities

Table 6 – Perceived usefulness of ArchConcept - statements 1 to 11.

Statement	1	2	3	4	5	p
1. Using ArchConcept allows me to better manage my work activities.		4	1	8	4	12
2. Using ArchConcept increases my performance at work.		4	5	5	3	8
3. In ArchConcept, the functionalities for registering, editing and deleting stakeholders are useful for my work.		3	4	4	6	10
4. In ArchConcept, the functionality of adding and deleting concerns is useful for my work.		3	2	7	5	12
5. In ArchConcept, the functionalities for registering, editing and deleting viewpoints are useful for my work.		3	2	7	5	12
6. In ArchConcept, the problem registration, editing and deletion functionalities are useful for my work.		2	4	6	5	11
7. In ArchConcept, the functionalities for registering, editing and deleting objectives are useful for my work.	1	2	3	6	5	11
8. In ArchConcept, the functionalities for registering, editing and deleting decisions are useful for my work.	1	2	2	6	6	12
9. In ArchConcept, the functionalities for registering, editing and deleting tradeoffs are useful for my work.	1	3	1	6	6	12
10. ArchConcept enables me to perform my tasks in less time.		5	4	7	1	8
11. Using ArchConcept increases my productivity.		5	4	6	2	8

of the ArchConcept in their work. Only 8 answered that ArchConcept is useful in their work.

4.4.3 Threats to validity

This research presents some threats to validity that range from internal, construct, statistical, and external validity. This section discusses the strategies used to manage these threats.

Internal validity.

Construct. Due to the Covid-19 pandemic, it was not possible to have face-to-face training before the use of the ArchConcept. The participants read the user guide, used the application, and answered the questionnaire. It might have some misunderstandings in this process. The number of respondents may not be enough to generalize the findings, although they are experienced practitioners and researchers.

Statistical. The objective was to find the representative power of the results, and not necessarily the statistical significance.

It should be emphasized that perceived usefulness and ease of use are people's subjective appraisal of performance and effort, respectively, and do not necessarily reflect objective reality (DAVIS, 1989). Though different individuals may attribute a slightly different meaning to particular statements, the goal of the multi-item approach is to reduce any extraneous effects of individual items.

Table 7 – Perceived ease of use of ArchConcept - statements 1 to 17.

Statement	**1**	**2**	**3**	**4**	**5**	**p**
1. It is easy for me to register, edit or delete stakeholders.			1	5	11	16
2. It is easy for me to correlate stakeholders and concerns.		1		8	8	16
3. For me, it is easy to register, edit or delete viewpoints.			1	8	8	16
4. For me, it is easy to register a viewpoint from the concern already registered.			1	9	7	16
5. For me, it is easy to add, edit or delete problems.		1		7	9	16
6. It is easy for me to set up a problem, associating it to one or more stakeholders.			1	9	7	16
7. It is easy for me to add, edit or delete objectives.	1		2	9	5	14
8. For me, it is easy to register one or more objectives from an already registered problem.	1		4	7	5	12
9. For me, it is easy to add, edit, or delete decisions.	1	1	2	7	6	13
10. For me, it is easy to add one or more decisions from a previously registered objective.	1	2	1	7	6	13
11. It is easy for me to add, edit, or delete tradeoffs.	1	3	1	8	4	12
12. For me, it is easy to register one or more tradeoffs from a previously registered decision.	1	2	2	8	4	12
13. It is easy for me to generate architecture conceptualization reports.		1		6	10	16
14. It is easy for me to remember how to execute tasks in ArchConcept.		1	1	6	9	15
15. I can use ArchConcept without consulting the user manual.	1	3	3	7	3	10
16. Using ArchConcept makes my work easier to do.		3	3	9	2	11
17. It was easy for me to use ArchConcept.		1	1	11	4	15

Table 8 – Perceived usage of ArchConcept - statements 1 to 5.

Statement	**1**	**2**	**3**	**4**	**5**	**p**
1. I am interested in knowing more about the Architecture Conceptualization process, present in the ISO/IEC/IEEE 42020 Standard.		2	2	8	5	13
2. The reports generated by ArchConcept are useful for me.	1	3	3	6	4	10
3. ArchConcept is useful for my work.		4	5	5	3	8
4. I would use at least one of ArchConcept's functionalities in my work.	1	3	1	7	5	12
5. I would use most or all of the functionality of ArchConcept in my work.	3	4		7	3	10

In this research, we are interested in the individuals' acceptance process, which is a good starting point to view how new technology might be accepted in a community. Therefore, extensions to TAM in which the social context is taken into account, although useful, are not investigated here. In addition, the purpose is to evaluate the initial acceptance of technology, which makes TAM a suitable model.

External validity. It refers to the validity of the obtained results in other wider contexts. Since this study has not been replicated yet, we describe some criteria that can be used to identify similar contexts where our findings can be closely applied to: the participants should be from areas related to Software Engineering and Software Architecture, practicing work as Researchers, System Analysts, University Professors, and Project Managers, with some experience in Software Developing and Software Architecture.

Thus, we can infer that several aspects of the research can be found in other contexts, although the results cannot be generalized, needing that new experimental studies can be carried out.

5 Conclusion

Although the architecture processes can be executed simultaneously with the interactions between them and the iteration over time, as with the Core Processes of ISO/IEC/IEEE 42020 (Architecture Conceptualization, Architecture Evaluation, and Architecture Elaboration), this document dealt with Conceptualization activities of software architecture. Following the activities and tasks in Clause 8 helps to start the architectural effort to clarify stakeholders' needs.

On the other hand, some of the tasks seem to be repeated. For example, task "a) Identify the general nature of the problem area(s) that needs to be addressed" of activity "8.4.1 Prepare for and plan the architecture conceptualization effort", and task "a) Identify the potential problem area(s) that needs to be addressed" of activity "8.4.3 Characterize problem space". This is likely because the processes are interrelated, but the architecture team needs to be aware of these situations.

In addition, considering the high number of tasks (100), and their high level of abstraction, understanding and using the Architecture Conceptualization clause in practice, in industry, seems to be a challenge. Considering the novelty of the ISO/IEC/IEEE 42020 Standard, tools such as ArchConcept are the first step towards implementation of the guidelines of that Standard so that the industry can evaluate the benefits.

Given the relevance of ISO/IEC/IEEE 42020 to provide useful and mature guidelines in terms of the Software Architecture Process, it is important that organizations may consider adopting it in the near future.

The Standard ISO/IEC/IEEE 42020 presents 6 Architecture processes with high level of abstraction in many of their Activities. It offers possible integration with related Standards such as ISO/IEC/IEEE 42010, which presents a way to create an Architecture Description. ISO/IEC/IEEE 42010 proposes the system software architecture as a product composed of models, views, viewpoints, decisions, and other architectural elements, which represent important entities of software architecture, but are at a lower level of abstraction when compared to the framework proposed in this article.

For instance, task “l) Develop an architecture description consisting of relevant viewpoints, views, models, model correspondences and express them in the specified form with a level of detail, correctness, and completeness suitable for their intended use” of activity “8.4.8 Capture architecture concepts and properties”. In other words, ISO/IEC/IEEE 42020 provides processes for the application of Architecture Description defined by ISO/IEC/IEEE 42010 (ISO/IEC/IEEE, 2019a).

The work carried out proposed a framework and tool to help in organizing the activities and tasks proposed in Clause 8 of ISO/IEC/IEEE 42020, leading to a mature characterization of the solution proposed by the architecture. The ArchConcept was designed to address the high-level abstraction of the Standard ISO/IEC/IEEE 42020, and can be useful for software architects that want to follow ISO/IEC/IEEE 42020’s recommendation and achieve high-quality results in their work of software architecture conceptualization. In the software industry, there would be many solutions, and the interested parties could decide on the best one according to their needs. Therefore, a software tool for architecture conceptualization is useful.

A web-based application was designed, developed, and evaluated to support software architects in using the activities and tasks of the Architecture Conceptualization clause based on the framework proposed. The Archconcept implemented activities 1, 3, 5, 6, 7 and 8 of the framework. Activities 2, 4, 9 and 10 were not contemplated as the application is experimental, covering main aspects regarding the Architecture Conceptualization and would require higher complexity.

As ArchConcept is focused on the early stages of the project (Architecture Conceptualization), the results found in this work could be evidence of the short time dedicated to the initial phase of projects and their consequences, like misunderstandings. Some respondents answered they do not perceive the usefulness of the ArchConcept system because they did not realize the importance of the Architecture Conceptualization to the system, or because there is a lack of knowledge, as 13 respondents were interested in knowing more about the Architecture Conceptualization process.

As the ISO/IEC/IEEE 42020 Standard is new and still not well known in industry and academia, the results indicated that the web-based application was not properly perceived in terms of usefulness and usage, although many respondents realized it just by reading the user guide. On the other hand, the application was considered easy for daily use.

5.1 Results

The results obtained in this master’s dissertation present some practical contributions and some contributions to the research itself. Thus, the following contributions resulting from this research are highlighted below:

- Framework: A framework was proposed to be used by professionals who want to get to know an example of the adoption of the Clause 8 (Conceptualization Process) of the Standard ISO/IEC/IEEE 42020.
- ArchConcept: A web system was implemented and evaluated and can be useful for software architects that want to follow the ISO/IEC/IEEE 42020's recommendation and achieve high-quality results in their work of software architecture conceptualization.

5.2 Future works

This research work generated as its main result the ArchConcept. This web system could be used to support professionals, but needs further evolution to deliver more functionalities. Although the objectives have been achieved, there are still some suggestions and recommendations for future work for this research:

- The framework could be refined to explore more examples of the activities and tasks of the Architecture Conceptualization Process present in the ISO/IEC/IEEE 42020 standard. As the Architecture Conceptualization Process can be applied widely to many kinds of projects, demonstrating it through the framework could show to related personnel the applicability of adopting the ISO/IEC/IEEE 42020 standard in their teams.
- The web-based application to assist in the management of the activities and tasks of the Architecture Conceptualization Process needs more functionalities such as allowing multiple users to manage the project and export reports. As an experimental application, more functionalities can be used by software architects and eventually develop a professional version.
- Review the user guide to facilitate the use/adoption of the proposed web system. It may incentive people to use the application.
- An assessment by Software Architecture experts, both from academia and industry, on the web system. This assessment aims to obtain feedback about the advantages and disadvantages of the use of the proposed web system. It may help to identify if the application is suitable for one or both groups.

5.3 Thesis-Related Publications

5.3.1 Published Papers

Santos V.M., Misra S., Soares M.S. (2020) Architecture Conceptualization for Health Information Systems Using ISO/IEC/IEEE 42020. In: Gervasi O. et al. (eds) Computational Science and Its Applications – ICCSA 2020. ICCSA 2020. Lecture Notes in Computer Science, vol 12254. p.398-411. Springer, Cham. Qualis A3.

5.3.2 Submitted Papers

Design and Evaluation using Technology Acceptance Model of an Architecture Conceptualization Framework System based on ISO/IEC/IEEE 42020.

Bibliography

ABRAHAMSSON, P.; BABAR, M. A.; KRUCHTEN, P. Agility and Architecture: Can They Coexist? *IEEE Software*, v. 27, n. 2, p. 16–22, March 2010. Citado na página 13.

ADAMS, D. A.; NELSON, R. R.; TODD, P. A. Perceived Usefulness, Ease of Use, and Usage of Information Technology: a Replication. *MIS Quarterly*, v. 16, n. 2, p. 227–247, 1992. Citado na página 18.

ANSI/IEEE. ANSI/IEEE Std 1471 Recommended Practice for Architectural Description of Software-Intensive Systems. 2000. Citado na página 12.

ASCHER, D.; EHLERS, F. Seamless Integration of ATR Services into Situation-Adaptive Distributed Systems via Reference Modeling. In: *6th Underwater Acoustics Conference and Exhibition*. [S.l.]: ASA, 2021. Citado na página 15.

BOEHM, B. W. Megaprogramming. In: *Proceedings of the DARPA Software Technology Conference, Los Angeles, CA*. [S.l.]: DARPA, 1993. Citado na página 12.

BOEHM, B. W. et al. Software Requirements Negotiation and Renegotiation aids: A theory-W based spiral approach. In: PERRY, D. E.; JEFFERY, R.; NOTKIN, D. (Ed.). *17th International Conference on Software Engineering, Seattle, Washington, USA, April 23-30, 1995, Proceedings*. [S.l.]: ACM, 1995. p. 243–253. Citado na página 12.

BOOCH, G. The Economics of Architecture-First. *IEEE Software*, v. 24, n. 5, p. 18–20, 2007. Citado na página 13.

BOOCH, G. et al. *Object-Oriented Analysis and Design with Applications (3rd Edition)*. [S.l.]: Addison-Wesley Longman Publishing Co., Inc., 2007. Citado na página 14.

BROWN, A. W.; MCDERMID, J. A. The Art and Science of Software Architecture. *International Journal of Coorperative Information Systems*, v. 16, n. 3/4, p. 439–466, 2007. Citado na página 13.

BROY, M. The 'Grand Challenge' in Informatics: Engineering SoftwareIntensive Systems. Computer, v. 39, n. 10, p. 72–80, 2006. Citado na página 13.

CAPILLA, R. et al. 10 Years of Software Architecture Knowledge Management: Practice and Future. *Journal of Systems and Software*, v. 116, p. 191–205, 2016. Citado na página 34.

CLEMENTS, P. et al. *Documenting Software Architectures: Views and Beyond*. [S.l.]: Addison-Wesley Professional, 2002. Citado na página 13.

CREFF, S. et al. Towards Facilities for Modeling and Synthesis of Architectures for Resource Allocation Problem in Systems Engineering. In: LOPEZ-HERREJON, R. E. (Ed.). *SPLC '20: 24th ACM International Systems and Software Product Line Conference, Montreal, Quebec, Canada, October 19-23, 2020, Volume A*. [S.l.]: ACM, 2020. p. 32:1–32:11. Citado 2 vezes nas páginas 19 and 20.

DASANAYAKE, S. et al. Impact of Requirements Volatility on Software Architecture: How do Software Teams Keep up with Ever-Changing Requirements? *Journal of Software: Evolution and Process*, v. 31, n. 6, 2019. Citado na página 13.

DAVIS, F. D. Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology. *Management Information Systems*, v. 13, n. 3, p. 319–340, 1989. Citado 2 vezes nas páginas 18 and 55.

DIAMPOVESA, S.; HUBERT, A.; YVARS, P. Designing Physical Systems through a Model-Based Synthesis Approach. Example of a Li-ion Battery for Electrical Vehicles. *Computers in Industry*, v. 129, p. 103440, 2021. Citado 3 vezes nas páginas 19, 20, and 27.

FALESSI, D.; AL et. Decision-making Techniques for Software Architecture Design: A Comparative Survey. *ACM Computing Surveys*, v. 43, n. 4, p. 1–33, 2011. Citado na página 15.

FALESSI, D. et al. Applying Empirical Software Engineering to Software Architecture: Challenges and Lessons Learned. *Empirical Software Engineering*, v. 15, n. 3, p. 250–276, 2010. Citado na página 12.

FRANCA, J.; LIMA, J. de S.; SOARES, M. S. Development of an Electronic Health Record Application using a Multiple View Service Oriented Architecture. In: *ICEIS 2017 - 19th International Conference on Enterprise Information Systems, Proceedings*. [S.l.: s.n.], 2017. p. 308–315. Citado na página 37.

FRANCA, J.; SOARES, M. S. SOAQM: Quality model for SOA applications based on ISO 25010. In: *ICEIS 2017 - 19th International Conference on Enterprise Information Systems, Proceedings*. [S.l.: s.n.], 2015. v. 2. Citado na página 37.

GALSTER, M.; TAMBURRI, D. A.; KAZMAN, R. Towards Understanding the Social and Organizational Dimensions of Software Architecting. *ACM SIGSOFT Software Engineering Notes*, Association for Computing Machinery, v. 42, n. 3, p. 24–25, 2017. Citado na página 13.

GARLAN, D. Software Architecture: a Roadmap. In: FINKELSTEIN, A. (Ed.). *22nd International Conference on on Software Engineering, Future of Software Engineering Track, ICSE 2000, Limerick Ireland, June 4-11, 2000*. [S.l.]: ACM, 2000. p. 91–101. Citado na página 12.

GARLAN, D. Software Architecture: A Travelogue. In: HERBSLEB, J. D.; DWYER, M. B. (Ed.). *Proceedings of the on Future of Software Engineering, FOSE 2014, Hyderabad, India, May 31 - June 7, 2014*. [S.l.]: ACM, 2014. p. 29–39. Citado 2 vezes nas páginas 12 and 13.

GARLAN, D.; SHAW, M. A Introduction to Software Architecture. In: AMBRIOLA, V.; TORTORA, G. (Ed.). *Advances in Software Engineering and Knowledge Engineering*. [S.l.]: World Scientific, 1993, (Series on Software Engineering and Knowledge Engineering, v. 2). p. 1–39. Citado na página 12.

GIL, A. *Como Elaborar Projetos de Pesquisa*. [S.l.]: Atlas, 2002. Citado na página 17.

HEESCH, U. van; AVGERIOU, P.; HILLIARD, R. A Documentation Framework for Architecture Decisions. *Journal of Systems and Software*, v. 85, n. 4, p. 795–820, 2012. Citado 2 vezes nas páginas 16 and 18.

HODA, R.; SALLEH, N.; GRUNDY, J. C. The Rise and Evolution of Agile Software Development. *IEEE Software*, v. 35, n. 5, p. 58–63, 2018. Citado na página 13.

IEEE. IEEE Standard for Application and Management of the Systems Engineering Process. *IEEE 1220:2005 (IEEE Standard for Application and Management of the Systems Engineering Process)*, p. 1–97, Sep 2005. Citado na página 26.

ISO. ISO Enterprise Modelling and Architecture — Requirements for Enterprise-referencing Architectures and Methodologies. *15704:2019(E)*, p. 1–70, Dec 2019. Citado na página 21.

ISO/IEC. ISO/IEC Systems and software engineering — Software Life Cycle Processes. *12207:2017(E)*, p. 1–138, Nov 2017. Citado na página 21.

ISO/IEC/IEEE. *ISO/IEC/IEEE Information Technology — Security techniques — Verification of cryptographic protocols*, 2011. Citado na página 34.

ISO/IEC/IEEE. Systems and Software Engineering - Architecture Description. *ISO/IEC/IEEE 42010:2011(E) (Revision of ISO/IEC 42010:2007 and IEEE Std 1471-2000)*, p. 1–46, Dec 2011. Citado 2 vezes nas páginas 14 and 41.

ISO/IEC/IEEE. ISO/IEC/IEEE International Standard - Systems and software engineering - System Life Cycle Processes. *15288:2015(E)*, p. 1–118, May 2015. Citado 3 vezes nas páginas 19, 21, and 33.

ISO/IEC/IEEE. ISO/IEC/IEEE International Standard - Systems and Software Engineering-Vocabulary. *24765:2017(E)*, p. 1–541, Aug 2017. Citado na página 29.

ISO/IEC/IEEE. ISO/IEC/IEEE International Standard - Software, Systems and Enterprise - Architecture Processes. *42020:2019(E)*, p. 1–126, July 2019. Citado 11 vezes nas páginas 14, 16, 18, 21, 22, 26, 27, 29, 32, 34, and 59.

ISO/IEC/IEEE. Software, Systems and Enterprise — Architecture Evaluation Framework. *42030:2019(E)*, p. 1–77, July 2019. Citado na página 14.

JOHANNESSON, P.; PERJONS, E. *An Introduction to Design Science*. [S.l.]: Springer, 2014. Citado na página 17.

JONES, A. K. The Maturing of Software Architecture. In: *Software Engineering Symposium, Software Engineering Institute, Pittsburgh, Pa., August, 1994*. [S.l.]: Software Engineering Institute, 1994. Citado na página 12.

JUNIOR, A. A. da C.; MISRA, S.; SOARES, M. S. A Systematic Mapping Study on Software Architectures Description Based on ISO/IEC/IEEE 42010: 2011. In: MISRA, S. et al. (Ed.). *Computational Science and Its Applications - ICCSA 2019 - 19th International Conference, Saint Petersburg, Russia, July 1-4, 2019, Proceedings, Part V*. [S.l.]: Springer, 2019. (Lecture Notes in Computer Science, v. 11623), p. 17–30. Citado na página 13.

KNODEL, J.; NAAB, M. Software Architecture Evaluation in Practice: Retrospective on More Than 50 Architecture Evaluations in Industry. In: *2014 IEEE/IFIP Conference on Software Architecture*. [S.l.: s.n.], 2014. p. 115–124. Citado na página 13.

KRUCHTEN, P. *The Rational Unified Process: An Introduction*. [S.l.]: Addison-Wesley Longman Publishing Co., Inc., 2003. Citado na página 13.

KRUCHTEN, P.; OBBINK, J. H.; STAFFORD, J. A. The Past, Present, and Future for Software Architecture. *IEEE Software*, v. 23, n. 2, p. 22–30, 2006. Citado 2 vezes nas páginas 12 and 15.

KUMAR, A. et al. Architecting Disruptive Digital Product-Service Systems. *INCOSE International Symposium*, v. 28, p. 1280–1295, 07 2018. Citado 2 vezes nas páginas 19 and 20.

LIKERT, R. A Technique for the Measurement of Attitudes. *Archives of Psychology*, p. 1–55, 1932. Citado na página 54.

LINDGREN, M.; NORSTROM, C.; WALL A. LAND, R. Importance of software architecture during release planning. In: *Seventh Working IEEE / IFIP Conference on Software Architecture (WICSA 2008), 18-22 February 2008, Vancouver, BC, Canada*. [S.l.]: IEEE Computer Society, 2008. p. 253–256. Citado na página 15.

MARTIN, J. N. Overview of an Emerging Standard on Architecture Processes - ISO/IEC/IEEE 42020. In: *2018 Annual IEEE International Systems Conference, SysCon 2018, Vancouver, BC, Canada, April 23-26, 2018*. [S.l.]: IEEE, 2018. p. 1–8. Citado 2 vezes nas páginas 19 and 20.

MCLAIN, K. B.; O'HARA-LESLIE, E. K.; WADE, A. C. *Foundations for Assisting in Home Care*. [S.l.]: Open Suny Textbooks, 2016. Citado na página 35.

OLUMOFIN, F. G.; MISIC, V. B. Extending the ATAM Architecture Evaluation to Product Line Architectures. In: *5th Working IEEE/IFIP Conference on Software Architecture (WICSA'05)*. [S.l.: s.n.], 2005. p. 45–56. Citado na página 30.

OLUMOFIN, F. G.; MISIC, V. B. A Holistic Architecture Assessment Method for Software Product Lines. *Information and Software Technology*, v. 49, n. 4, p. 309–323, 2007. Citado na página 30.

PACE, J. A. D. et al. Producing Just Enough Documentation: An Optimization Approach Applied to the Software Architecture Domain. *Journal on Data Semantics*, v. 5, n. 1, p. 37–53, 2016. Citado na página 13.

PARETO, L. et al. Collaborative Prioritization of Architectural Concerns. *Journal of Systems and Software*, v. 85, n. 9, p. 1971–1994, 2012. Citado 2 vezes nas páginas 16 and 18.

PATTON, M. Q. Qualitative Research. In: *Encyclopedia of Statistics in Behavioral Science*. [S.l.]: American Cancer Society, 2005. Citado na página 17.

PERRY, D. E.; WOLF, A. L. Foundations for the study of software architecture. *ACM SIGSOFT Software Engineering Notes*, v. 17, n. 4, p. 40–52, 1992. Citado na página 12.

PHILLIPS, D. M.; MAZZUCHI, T. A.; SARKANI, S. An Architecture, System Engineering, and Acquisition Approach for Space System Software Resiliency. *Information and Software Technology*, v. 94, p. 150–164, 2018. Citado 2 vezes nas páginas 19 and 20.

PRESSMAN, R. S.; MAXIM, B. *Software Engineering: A Practitioner's Approach*. [S.l.]: McGraw-Hill Education, 2014. Citado na página 15.

QUEZADA-GAIBOR, D. et al. Cloud Platforms for Context-Adaptive Positioning and Localisation in GNSS-Denied Scenarios - A Systematic Review. *Sensors*, v. 22, n. 1, p. 110, 2022. Citado 2 vezes nas páginas 15 and 20.

SANTOS, R. S. *Metodologia Científca: A Construção do Conhecimento*. [S.l.]: Lamparina, 2007. Citado na página 17.

SANTOS, V. M.; MISRA, S.; SOARES, M. S. Architecture Conceptualization for Health Information Systems Using ISO/IEC/IEEE 42020. In: GERVASI, O. et al. (Ed.). *Computational Science and Its Applications - ICCSA 2020 - 20th International Conference, Cagliari, Italy, July 1-4, 2020, Proceedings, Part VI*. [S.l.]: Springer, 2020. (Lecture Notes in Computer Science, v. 12254), p. 398–411. Citado na página 15.

SOARES, M. *Architecture-Driven Integration of Modeling Languages for the Design of Software-Intensive Systems*. Tese (Doutorado) — Delft University of Technology, Netherlands, 2010. Citado 2 vezes nas páginas 13 and 14.

TIAKO, P. F. *Designing Software-Intensive Systems: Methods and Principles, 1st Edition*. [S.l.]: IGI Global, 2008. Citado na página 13.

ULUDAg, O.; MATTHES, F. Large-Scale Agile Development Patterns for Enterprise and Solution Architects. In: *Proceedings of the European Conference on Pattern Languages of Programs 2020*. New York, NY, USA: Association for Computing Machinery, 2020. (EuroPLoP '20). Citado na página 13.

VENKATESH, V.; BALA, H. Technology Acceptance Model 3 and a Research Agenda on Interventions. *Decision Sciences,*, v. 39, n. 2, p. 273–315, 2008. Citado na página 18.

WOODS, E. Software Architecture in a Changing World. *IEEE Software*, v. 33, p. 94–97, 11 2016. Citado na página 13.

YANG, C.; LIANG, P.; AVGERIOU, P. A Systematic Mapping Study on the Combination of Software Architecture and Agile Development. *Journal of Systems and Software*, v. 111, p. 157–184, 2016. Citado na página 13.

Appendix

APPENDIX A – ArchConcept User Guide

ArchConcept - Ferramenta para Conceitualização da Arquitetura de Software

Manual do Usuário

Sumário

1 - Introdução

A **ArchConcept** é uma ferramenta experimental baseada no Processo de Conceitualização da Arquitetura, integrante da Norma ISO/IEC/IEEE 42020.

O Processo de Conceitualização da Arquitetura (Clásula 8 da Norma ISO/IEC/IEEE 42020) tem como objetivos caracterizar o espaço do problema e determinar soluções que atendam os *concerns* dos *stakeholders*, definir os objetivos da arquitetura e atender aos requisitos relevantes.

Esse Processo apresenta muitas definições abstratas de alto nível e devido à grande quantidade de tarefas para fazer uma conceitualização de Arquitetura de Software baseada nessa Norma, o arquiteto de software despende muito tempo para entender o que é proposto, obter informações dos *stakeholders*, organizar e reunir as demais informações obtidas.

A **ArchConcept** ajuda o arquiteto de software a melhorar a produtividade e alcançar um resultado de alta qualidade, visto que a ferramenta é uma implementação das recomendações da Norma ISO/IEC/IEEE 42020, no tocante à Conceitualização da Arquitetura de Software.

De outra maneira, o usuário precisaria reunir essas informações em diversos documentos e planilhas. No caso de atualização de dados, seria gasto ainda mais tempo para procurar e efetivar as alterações.

2 - Definição de termos

A conceitualização da arquitetura de software tem como foco a identificação de soluções, com ênfase no entendimento completo de todo o espaço do problema.

A identificação do espaço do problema implica na definição e no estabelecimento dos objetivos da arquitetura, bem como na negociação com os principais interessados na priorização de seus *concerns*.

Os resultados da conceitualização da arquitetura são a definição do problema que está sendo tratado, o estabelecimento de objetivos arquiteturais que abordam os principais *concerns* dos *stakeholders*, a definição dos conceitos e propriedades chaves da arquitetura, e os princípios que orientam sua aplicação e evolução. Além desses resultados, trata dos principais *tradeoffs* e identifica possíveis limitações.

A **ArchConcept** registra dados cadastrados pelo usuário e gera relatórios a partir dessas informações, com a possibilidade de edição e exclusão de dados e a consequente atualização do relatório.

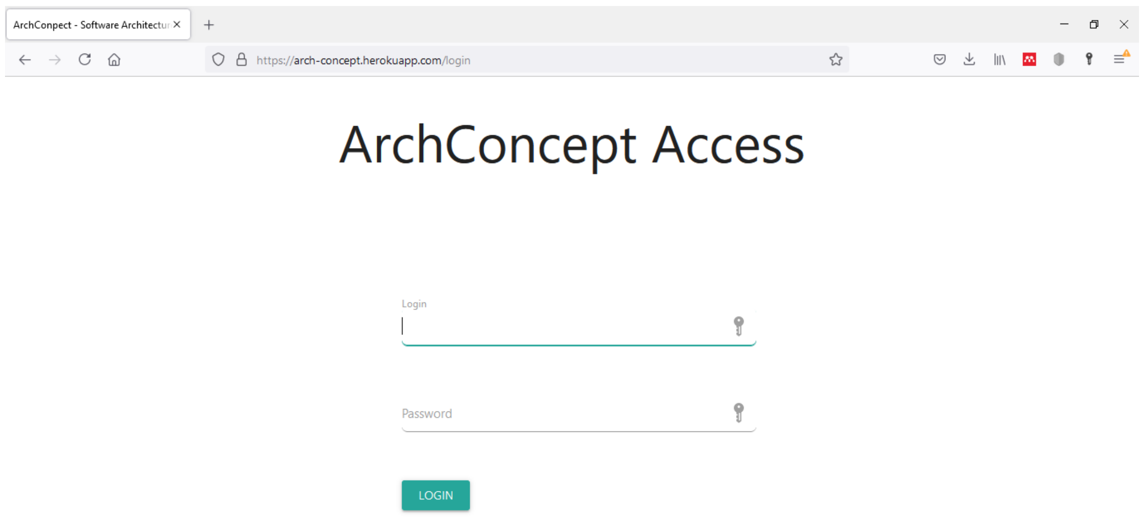
Os dados cadastrados são dos *stakeholders*, *concerns, viewpoints, problems, objectives, decisions* e *tradeoffs.* Não há impedimento para geração do relatório, caso não tenham sido cadastrados todos os dados.

- ***Stakeholder*** é qualquer indivíduo ou organização que tenha interesse no sistema ou em sua arquitetura.
- ***Concerns*** são interesses pertencentes ao desenvolvimento do software, sua operação e qualquer outro aspecto que é crítico ou importante para os *stakeholders*. Na **ArchConcept**, os *concerns* são cadastrados a partir do ponto de vista dos *stakeholders.* Uma descrição da arquitetura deve identificar os *stakeholders* do sistema tendo *concerns* considerados fundamentais para a arquitetura do sistema de interesse.
- ***Problems,*** o conceito de "problema" é usado no mesmo sentido da Norma ISO/IEC/IEEE 15288: "dificuldade, incerteza, ou evento realizado e indesejável, conjunto de eventos, condição ou situação que requer investigação e ação corretiva". Para caracterizar o problema, será necessário informar a área, os aspectos, os riscos, as oportunidades e as restrições.
- ***Objectives***, para cadastrar os objetivos identificados e definidos da arquitetura que abordam o(s) problema (s) e oportunidades com respeito aos *concerns* dos *stakeholders*.
- ***Decisions*** são decisões arquiteturais para atingir os objetivos da arquitetura.
- ***Tradeoffs*** para caracterizar as decisões. Tradeoffs entre os objetivos da arquitetura e as respectivas decisões e as limitações de viabilidade são cadastradas.
- ***Viewpoints*** com o objetivo de descrição da arquitetura, contendo os *concerns* que compõem o *viewpoint*, bem como a descrição dos modelos. Uma descrição da arquitetura deve incluir cada *viewpoint* da arquitetura usado nela. Cada *concern* identificado deve ser enquadrado por pelo menos um *viewpoint*

3 - Telas

3.1 - Tela de Login

Informar o Login e a Senha do usuário previamente fornecidos.

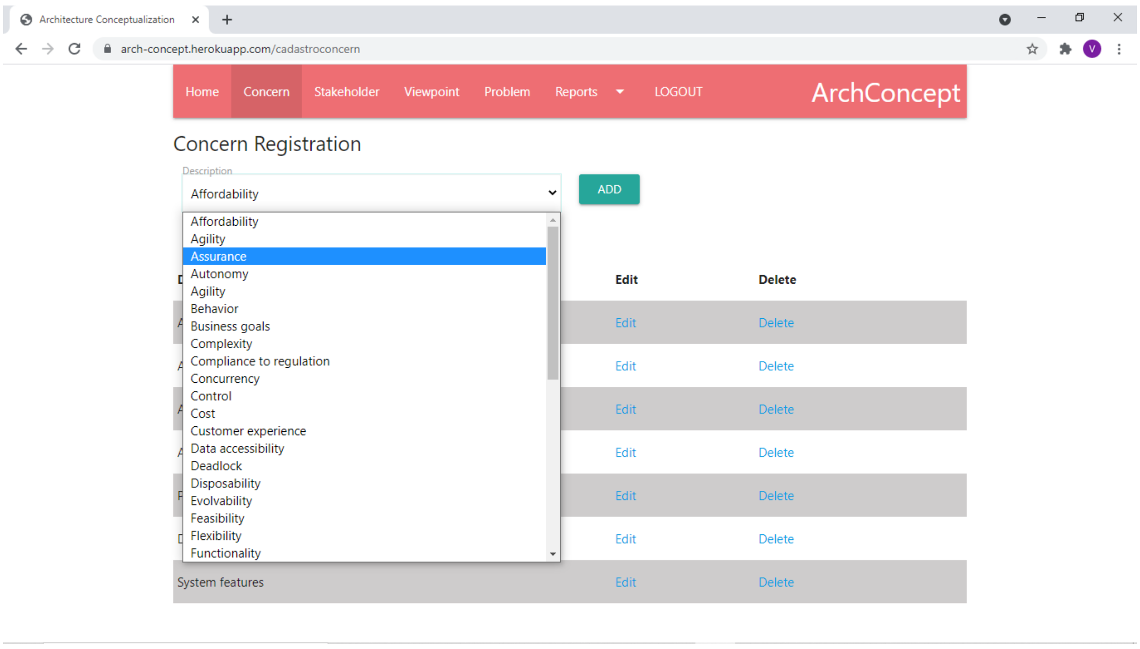

ArchConpect - Software Architectur
https://arch-concept.herokuapp.com/login
ArchConcept Access
Login
Password
LOGIN

3.2 - Tela de Concern

Selecionar a Descrição do *Concern*. Clicar em Add. Os dados cadastrados serão exibidos em tabela na parte inferior da tela.

As Descrições de *Concern* consideradas estão de acordo com a Norma ISO/IEC/IEEE 42010:

Functionality, Feasibility, Usage, System purposes, System features, System properties, Known limitations, Structure, Behavior, Performance, Resource utilization, Reliability, Security, Information assurance, Complexity, Evolvability, Openness, Concurrency, Autonomy, Cost, Schedule, Quality of service, Flexibility, Agility, Modifiability, Modularity, Control, Inter-process communication, Deadlock, State change, Subsystem Integration, Data accessibility, Privacy, Compliance to regulation, Assurance, Business goals and strategies, Customer experience, Maintainability, Affordability and Disposability.


Architecture Conceptualization

arch-concept.herokuapp.com/cadastroconcern

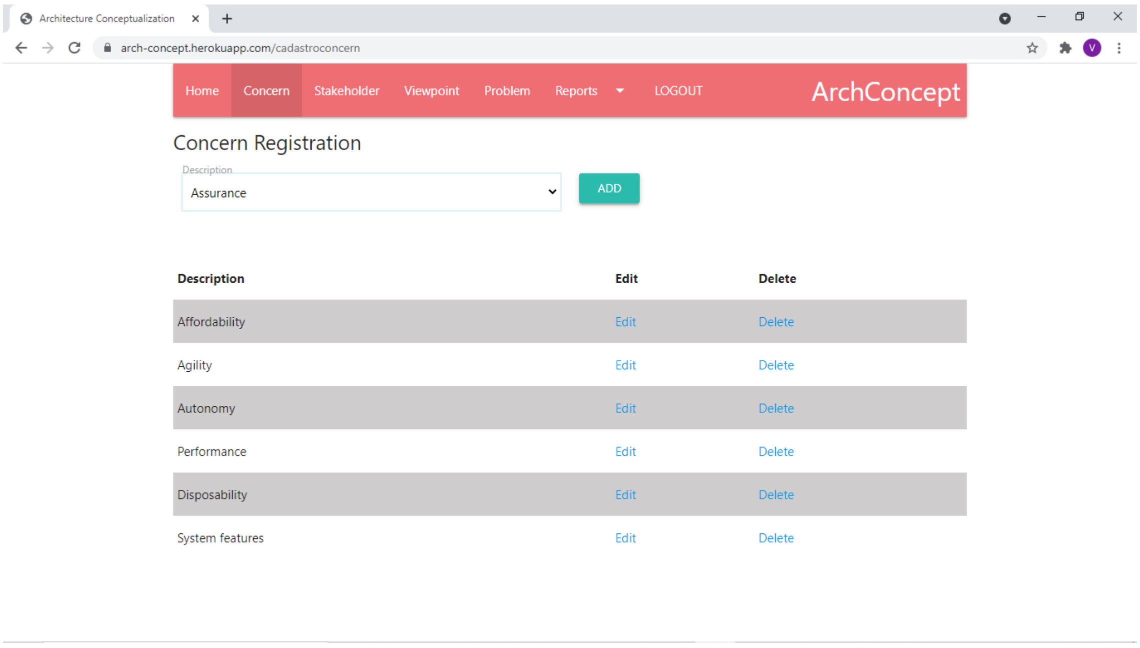
Home | Concern | Stakeholder | Viewpoint | Problem | Reports | LOGOUT | ArchConcept

Concern Registration

Description: Affordability — ADD

- Affordability
- Agility
- Assurance
- Autonomy
- Agility
- Behavior
- Business goals
- Complexity
- Compliance to regulation
- Concurrency
- Control
- Cost
- Customer experience
- Data accessibility
- Deadlock
- Disposability
- Evolvability
- Feasibility
- Flexibility
- Functionality

D	Edit	Delete
A	Edit	Delete
A	Edit	Delete
A	Edit	Delete
A	Edit	Delete
P	Edit	Delete
D	Edit	Delete
System features	Edit	Delete


Architecture Conceptualization

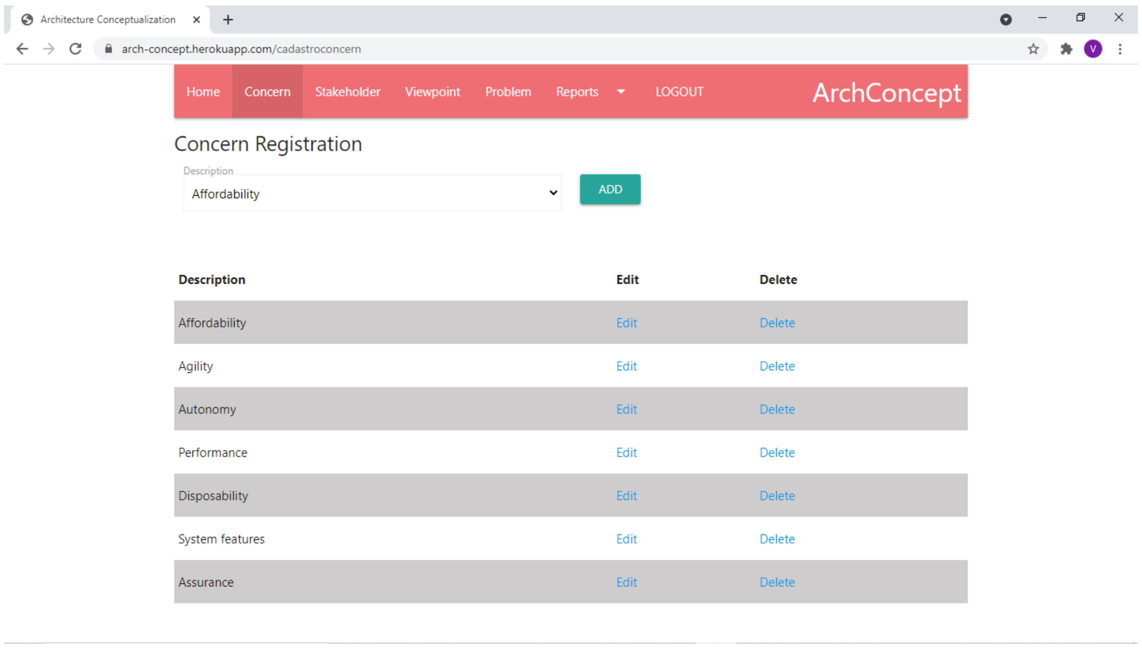
arch-concept.herokuapp.com/cadastroconcern

Home | Concern | Stakeholder | Viewpoint | Problem | Reports | LOGOUT | ArchConcept

Concern Registration

Description: Assurance — ADD

Description	Edit	Delete
Affordability	Edit	Delete
Agility	Edit	Delete
Autonomy	Edit	Delete
Performance	Edit	Delete
Disposability	Edit	Delete
System features	Edit	Delete


Architecture Conceptualization

arch-concept.herokuapp.com/cadastroconcern

Concern Registration

Description: Affordability

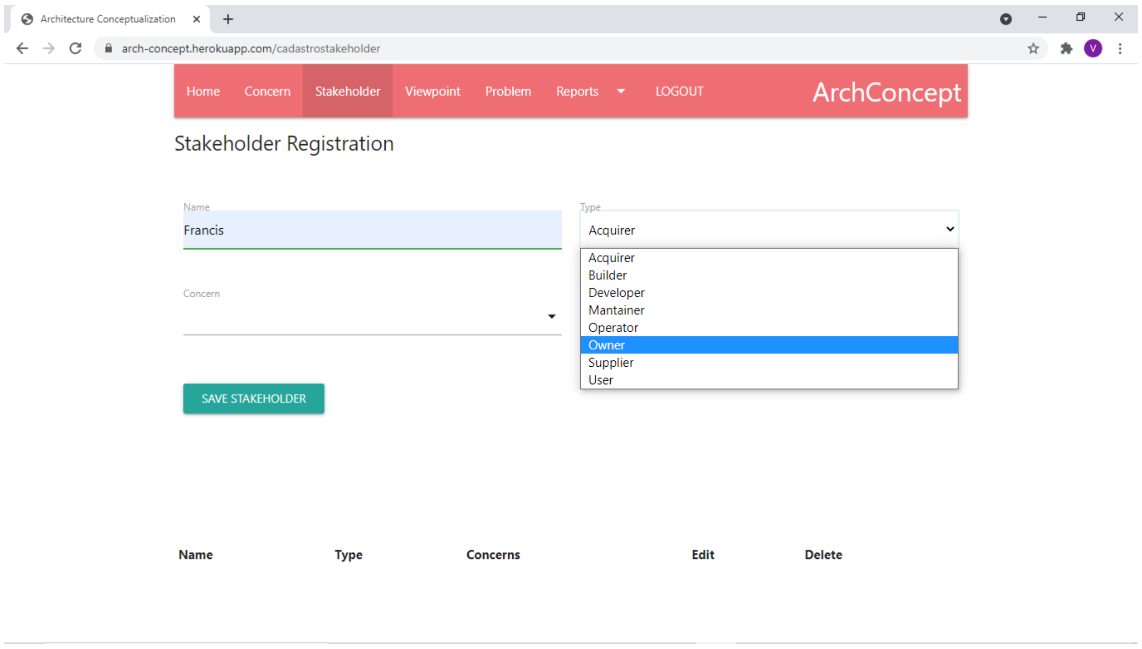
Description	Edit	Delete
Affordability	Edit	Delete
Agility	Edit	Delete
Autonomy	Edit	Delete
Performance	Edit	Delete
Disposability	Edit	Delete
System features	Edit	Delete
Assurance	Edit	Delete

3.3 - Tela de Stakeholder

Informar o Nome do *Stakeholder*, selecionar o Tipo e o Concern previamente cadastrado. Clicar em Save. Os dados cadastrados serão exibidos em tabela na parte inferior da tela.

Os seguintes tipos são considerados, de acordo com a Norma ISO/IEC/IEEE 42010:

User, Operator, Acquirer, Owner, Supplier, Developer, Builder of the system, Maintainer.


Architecture Conceptualization

arch-concept.herokuapp.com/cadastrostakeholder

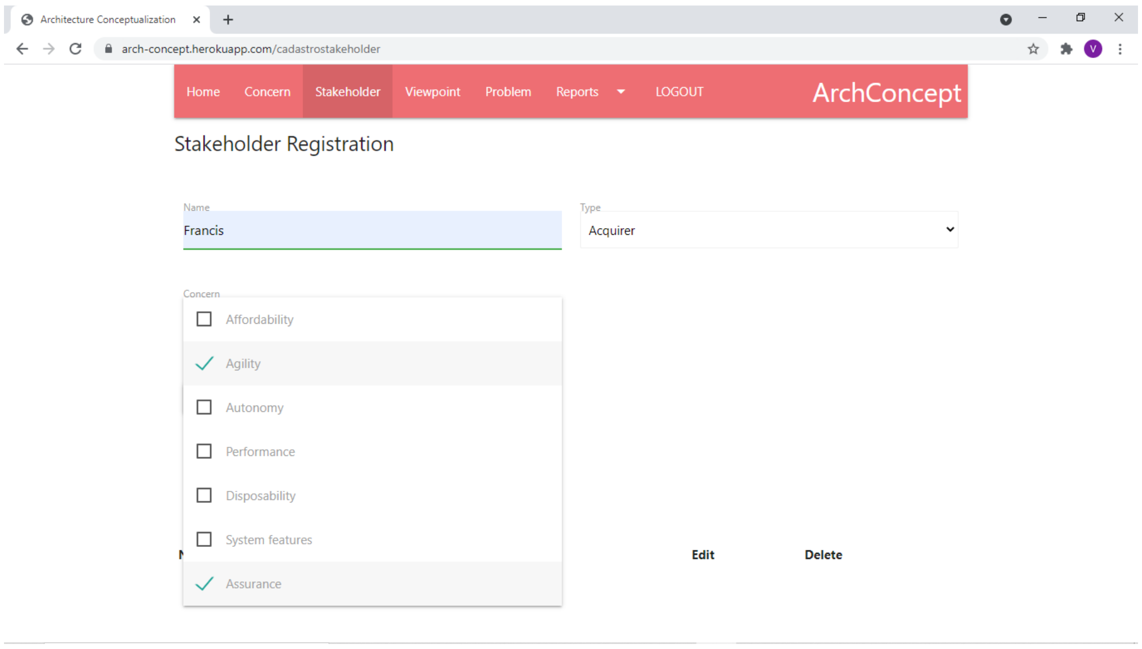
Stakeholder Registration

Name: Francis

Type: Acquirer

Concern:

Name	Type	Concerns	Edit	Delete


Architecture Conceptualization
arch-concept.herokuapp.com/cadastrostakeholder
Home
Concern
Stakeholder
Viewpoint
Problem
Reports
LOGOUT
ArchConcept

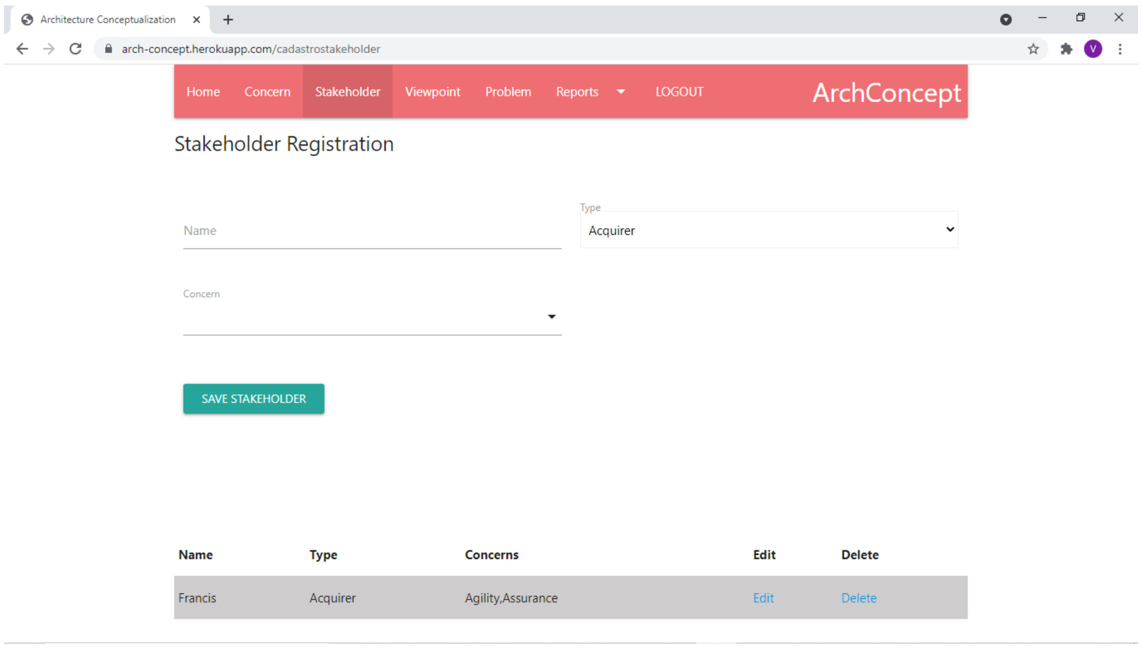
Stakeholder Registration

Name	Francis
Type	Acquirer

Concern

- [ ] Affordability
- ✓ Agility
- [ ] Autonomy
- [ ] Performance
- [ ] Disposability
- [ ] System features
- ✓ Assurance

N Edit Delete


Architecture Conceptualization
arch-concept.herokuapp.com/cadastrostakeholder
Home
Concern
Stakeholder
Viewpoint
Problem
Reports
LOGOUT
ArchConcept

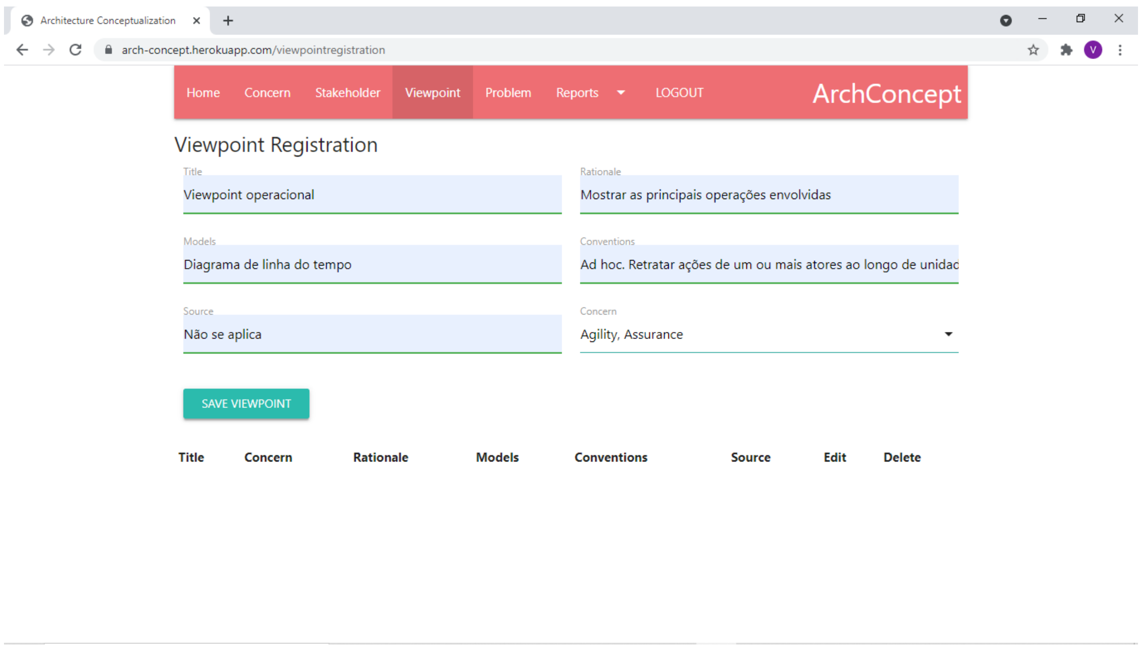
Stakeholder Registration

Name	
Type	Acquirer
Concern	

Name	Type	Concerns	Edit	Delete
Francis	Acquirer	Agility,Assurance	Edit	Delete

3.4 - Tela de Viewpoint

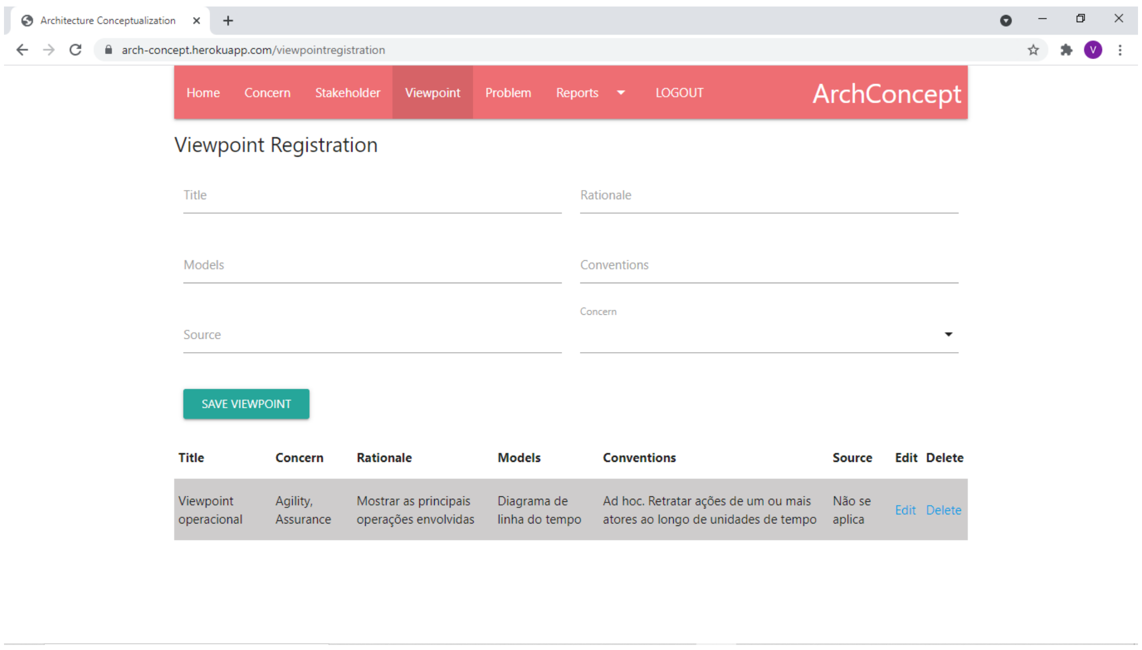
Informar o Título, Racional, Models, Conventions e Source do *Viewpoint* e selecionar um *Concern* (previamente cadastrado na tela de Concern). Clicar em Save. Os dados cadastrados serão exibidos em tabela na parte inferior da tela.


Architecture Conceptualization
arch-concept.herokuapp.com/viewpointregistration
Home
Concern
Stakeholder
Viewpoint
Problem
Reports
LOGOUT
ArchConcept
SAVE VIEWPOINT

Viewpoint Registration

Title: Viewpoint operacional	Rationale: Mostrar as principais operações envolvidas
Models: Diagrama de linha do tempo	Conventions: Ad hoc. Retratar ações de um ou mais atores ao longo de unidac
Source: Não se aplica	Concern: Agility, Assurance

Title	Concern	Rationale	Models	Conventions	Source	Edit	Delete

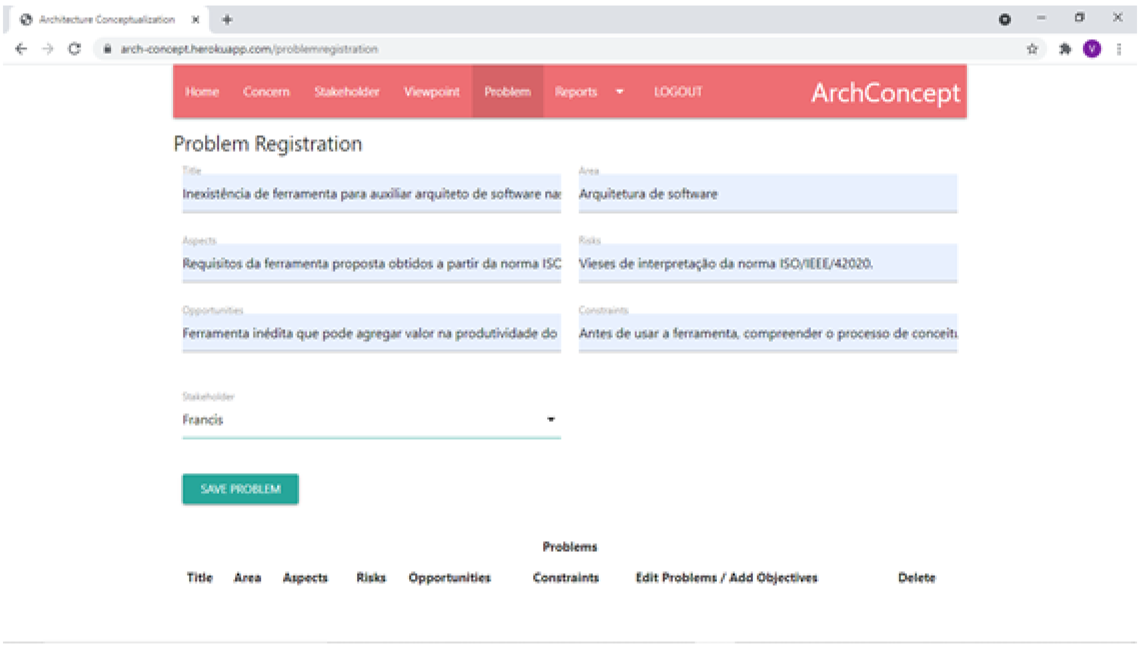


Viewpoint Registration

Title	Concern	Rationale	Models	Conventions	Source	Edit	Delete
Viewpoint operacional	Agility, Assurance	Mostrar as principais operações envolvidas	Diagrama de linha do tempo	Ad hoc. Retratar ações de um ou mais atores ao longo de unidades de tempo	Não se aplica	Edit	Delete

3.5 - Tela de Problema

Informar o Título, Área, Aspectos, Riscos, Oportunidades e Restrições do *Problem* e selecionar um ou mais *Stakeholders* (previamente cadastrado na tela de Stakeholder). Clicar em Save. Os dados cadastrados serão exibidos em tabela na parte inferior da tela.


Architecture Conceptualization
arch-concept.herokuapp.com/problemregistration
Home
Concern
Stakeholder
Viewpoint
Problem
Reports
LOGOUT
ArchConcept

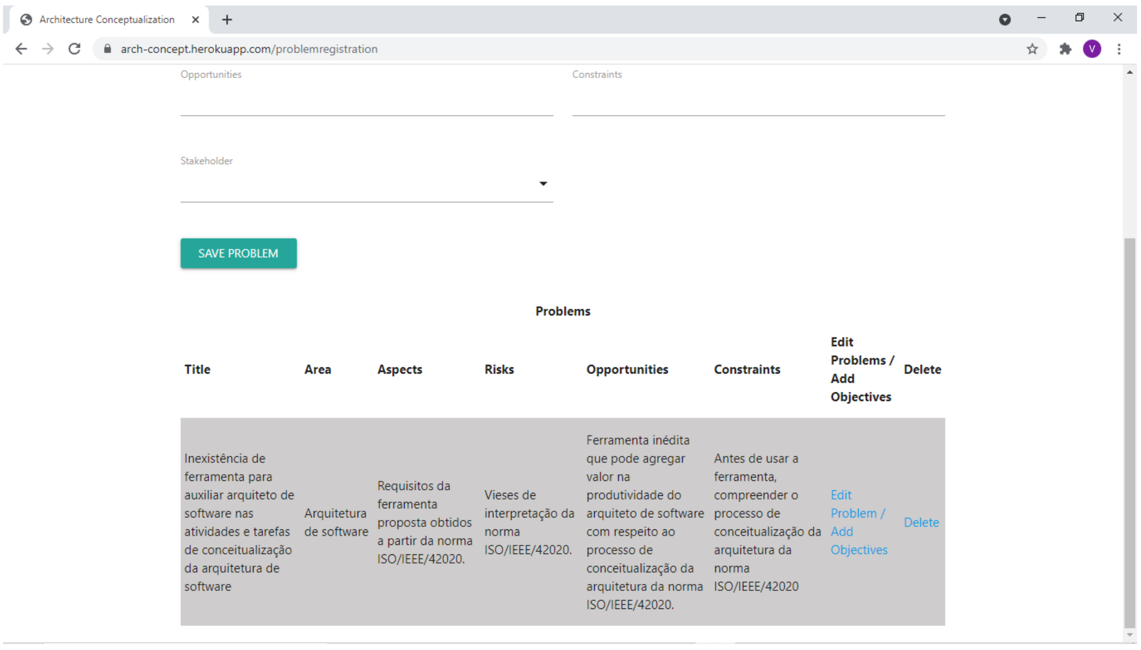
Problem Registration

Title: Inexistência de ferramenta para auxiliar arquiteto de software na	Area: Arquitetura de software
Aspects: Requisitos da ferramenta proposta obtidos a partir da norma ISC	Risks: Vieses de interpretação da norma ISO/IEEE/42020.
Opportunities: Ferramenta inédita que pode agregar valor na produtividade do	Constraints: Antes de usar a ferramenta, compreender o processo de conceitu
Stakeholder: Francis	

SAVE PROBLEM

Problems

Title	Area	Aspects	Risks	Opportunities	Constraints	Edit Problems / Add Objectives	Delete


Architecture Conceptualization
arch-concept.herokuapp.com/problemregistration

Opportunities

Constraints

Stakeholder

SAVE PROBLEM

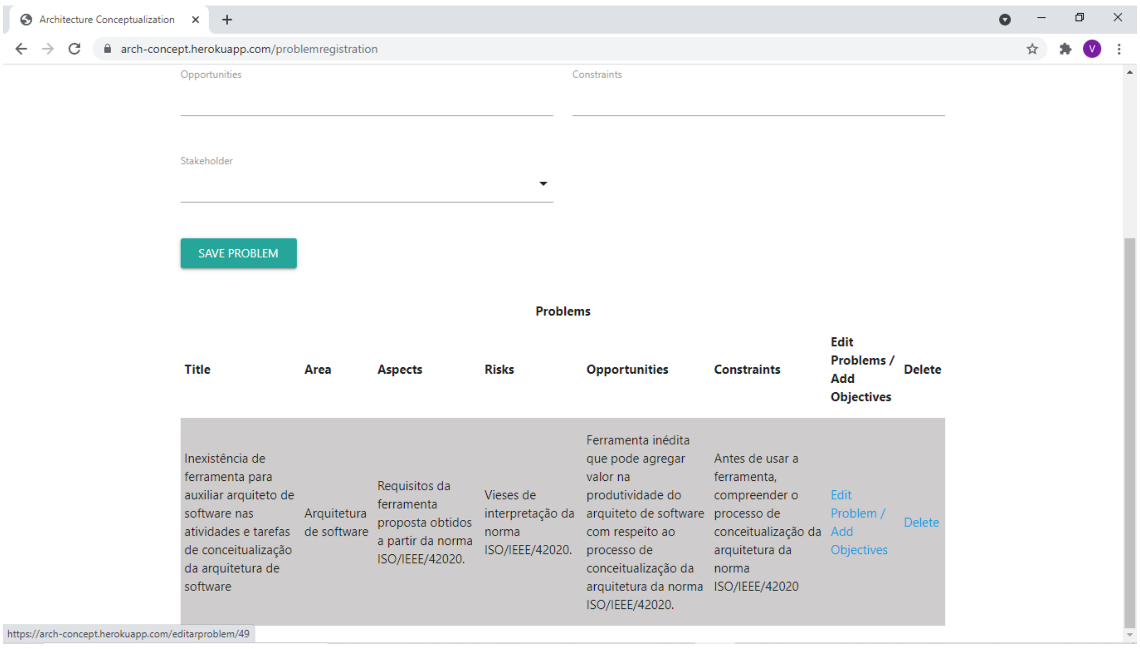
Problems

Title	Area	Aspects	Risks	Opportunities	Constraints	Edit Problems / Add Objectives	Delete
Inexistência de ferramenta para auxiliar arquiteto de software nas atividades e tarefas de conceitualização da arquitetura de software	Arquitetura de software	Requisitos da ferramenta proposta obtidos a partir da norma ISO/IEEE/42020.	Vieses de interpretação da norma ISO/IEEE/42020.	Ferramenta inédita que pode agregar valor na produtividade do arquiteto de software com respeito ao processo de conceitualização da arquitetura da norma ISO/IEEE/42020.	Antes de usar a ferramenta, compreender o processo de conceitualização da arquitetura da norma ISO/IEEE/42020	Edit Problem / Add Objectives	Delete

3.6 - Tela de Objetivo

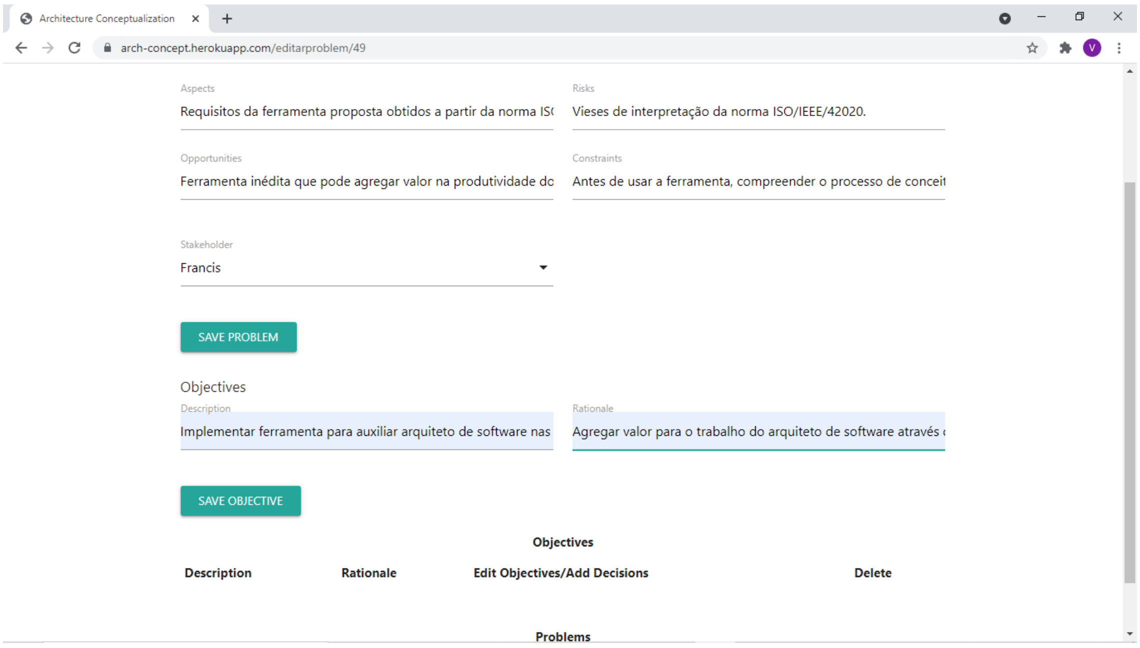
Clicar em "Add Objectives" na linha correspondente ao *Problem* que se deseja adicionar *Objectives*.

Informar Descrição e Racional do *Objective*. Clicar em Save. Os dados cadastrados serão exibidos em tabela na parte inferior da tela.


Architecture Conceptualization
arch-concept.herokuapp.com/problemregistration
Opportunities
Constraints
Stakeholder
SAVE PROBLEM
https://arch-concept.herokuapp.com/editarproblem/49

Problems

Title	Area	Aspects	Risks	Opportunities	Constraints	Edit Problems / Add Objectives	Delete
Inexistência de ferramenta para auxiliar arquiteto de software nas atividades e tarefas de conceitualização da arquitetura de software	Arquitetura de software	Requisitos da ferramenta proposta obtidos a partir da norma ISO/IEEE/42020.	Vieses de interpretação da norma ISO/IEEE/42020.	Ferramenta inédita que pode agregar valor na produtividade do arquiteto de software com respeito ao processo de conceitualização da arquitetura da norma ISO/IEEE/42020.	Antes de usar a ferramenta, compreender o processo de conceitualização da arquitetura da norma ISO/IEEE/42020	Edit Problem / Add Objectives	Delete


Architecture Conceptualization
arch-concept.herokuapp.com/editarproblem/49

Aspects: Requisitos da ferramenta proposta obtidos a partir da norma IS(

Risks: Vieses de interpretação da norma ISO/IEEE/42020.

Opportunities: Ferramenta inédita que pode agregar valor na produtividade dc

Constraints: Antes de usar a ferramenta, compreender o processo de conceit

Stakeholder: Francis

Objectives

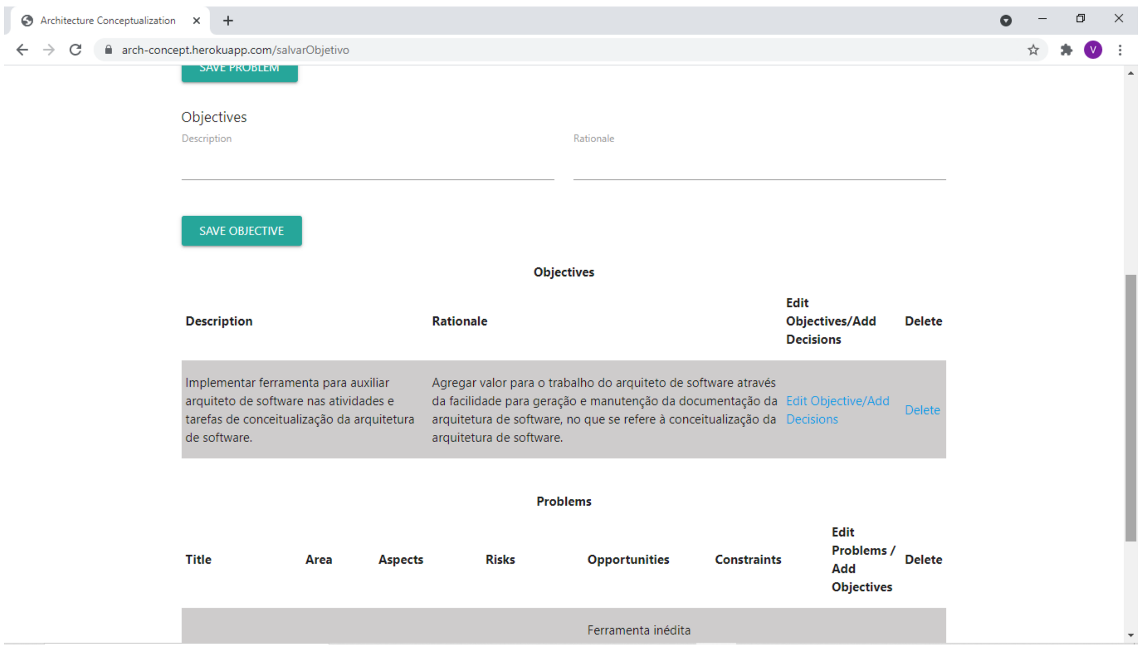
Description: Implementar ferramenta para auxiliar arquiteto de software nas

Rationale: Agregar valor para o trabalho do arquiteto de software através (

Objectives

Description	Rationale	Edit Objectives/Add Decisions	Delete

Problems


Architecture Conceptualization

arch-concept.herokuapp.com/salvarObjetivo

SAVE PROBLEM

Objectives

Description

Rationale

SAVE OBJECTIVE

Objectives

Description	Rationale	Edit Objectives/Add Decisions	Delete
Implementar ferramenta para auxiliar arquiteto de software nas atividades e tarefas de conceitualização da arquitetura de software.	Agregar valor para o trabalho do arquiteto de software através da facilidade para geração e manutenção da documentação da arquitetura de software, no que se refere à conceitualização da arquitetura de software.	Edit Objective/Add Decisions	Delete

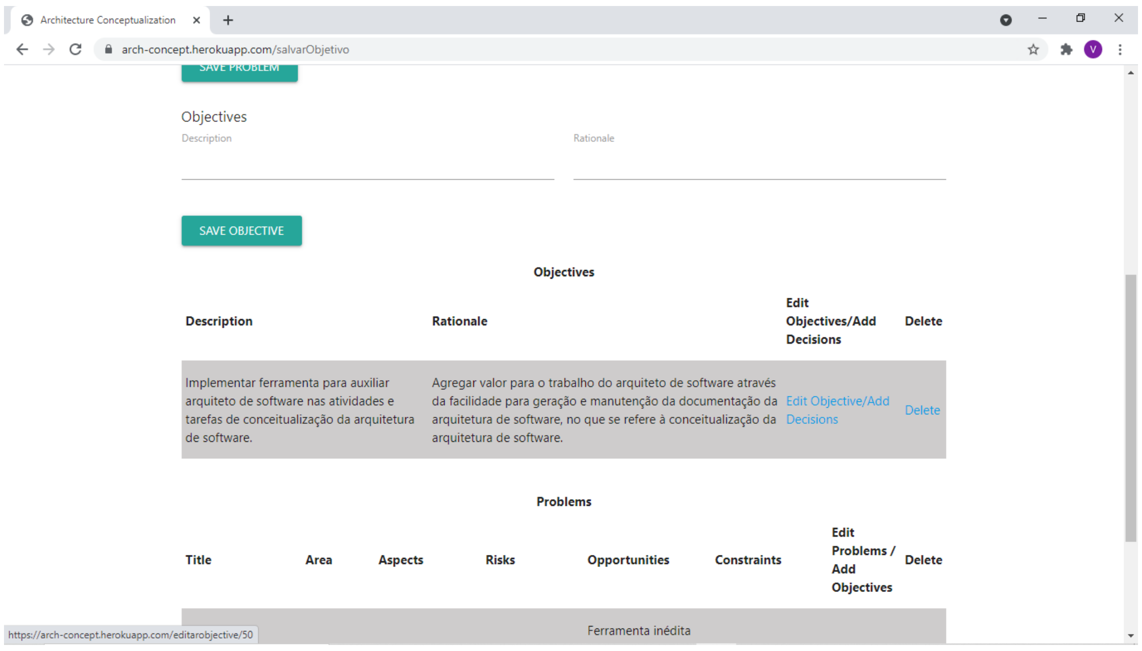
Problems

Title	Area	Aspects	Risks	Opportunities	Constraints	Edit Problems / Add Objectives	Delete
				Ferramenta inédita			

3.7 - Tela de Decisão

Clicar em “Add Decisions” na linha correspondente ao *Objective* que se deseja adicionar *Decisions*.

Informar Descrição, Solução e Racional da *Decision*. Clicar em Save. Os dados cadastrados serão exibidos em tabela na parte inferior da tela.



Architecture Conceptualization

arch-concept.herokuapp.com/salvarObjetivo

SAVE PROBLEM

Objectives

Description

Rationale

SAVE OBJECTIVE

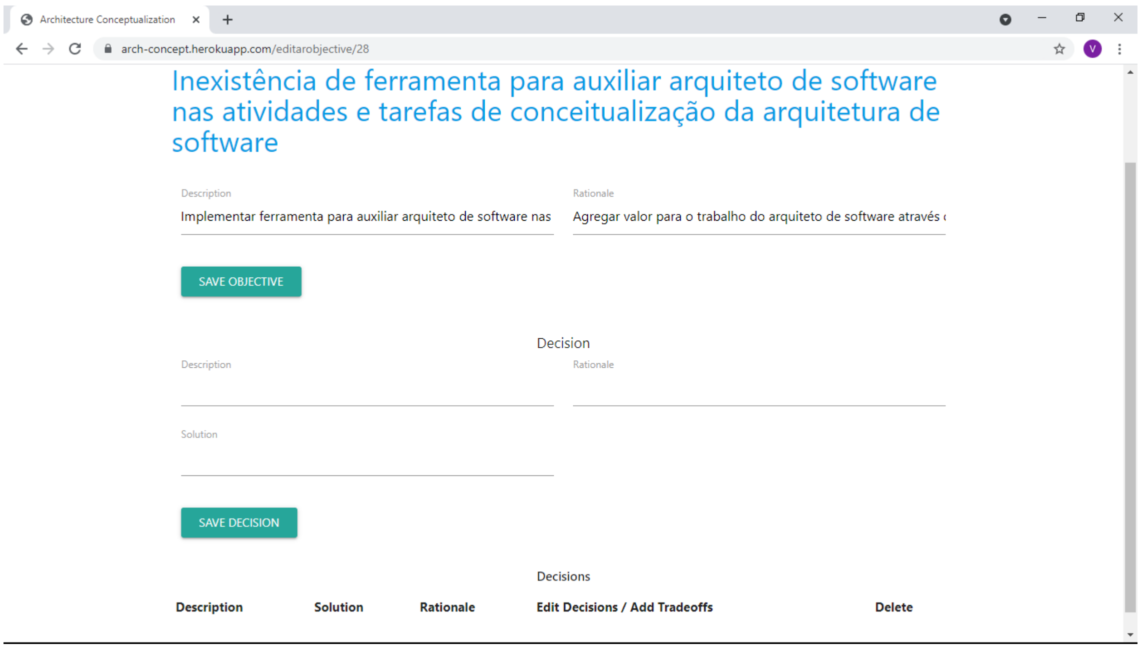
Objectives

Description	Rationale	Edit Objectives/Add Decisions	Delete
Implementar ferramenta para auxiliar arquiteto de software nas atividades e tarefas de conceitualização da arquitetura de software.	Agregar valor para o trabalho do arquiteto de software através da facilidade para geração e manutenção da documentação da arquitetura de software, no que se refere à conceitualização da arquitetura de software.	Edit Objective/Add Decisions	Delete

Problems

Title	Area	Aspects	Risks	Opportunities	Constraints	Edit Problems / Add Objectives	Delete
				Ferramenta inédita			

https://arch-concept.herokuapp.com/editarobjective/50



Architecture Conceptualization

arch-concept.herokuapp.com/editarobjective/28

Inexistência de ferramenta para auxiliar arquiteto de software nas atividades e tarefas de conceitualização da arquitetura de software

Description	Rationale
Implementar ferramenta para auxiliar arquiteto de software nas	Agregar valor para o trabalho do arquiteto de software através (

SAVE OBJECTIVE

Decision

Description

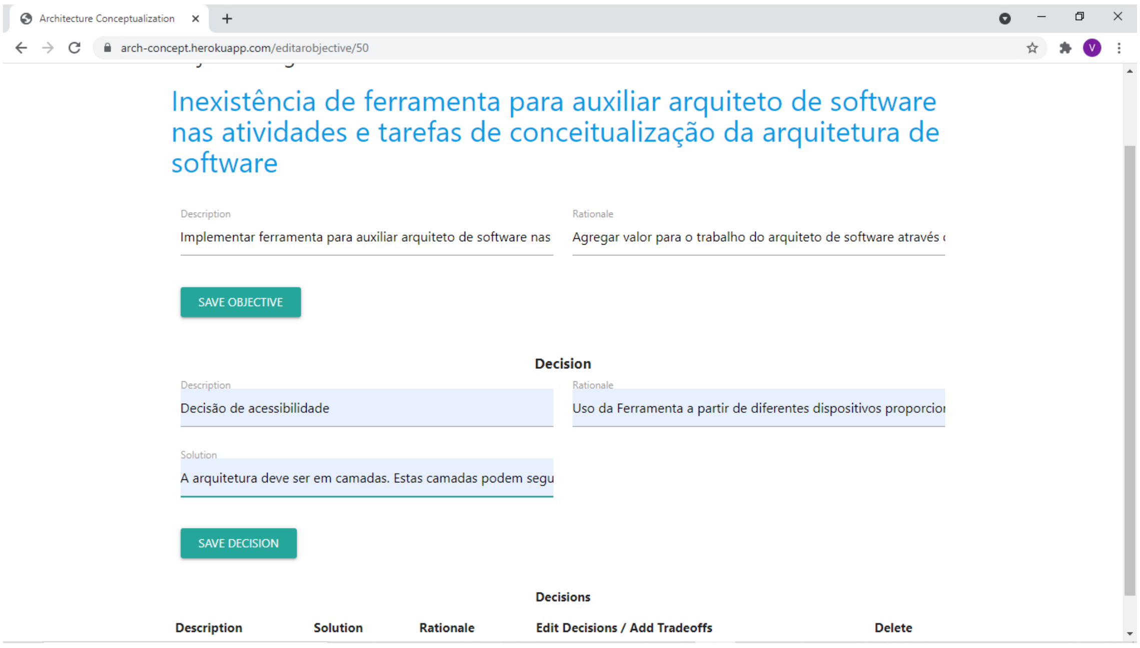
Rationale

Solution

SAVE DECISION

Decisions

Description	Solution	Rationale	Edit Decisions / Add Tradeoffs	Delete



Architecture Conceptualization

arch-concept.herokuapp.com/editarobjective/50

Inexistência de ferramenta para auxiliar arquiteto de software nas atividades e tarefas de conceitualização da arquitetura de software

Description	Rationale
Implementar ferramenta para auxiliar arquiteto de software nas	Agregar valor para o trabalho do arquiteto de software através (

SAVE OBJECTIVE

Decision

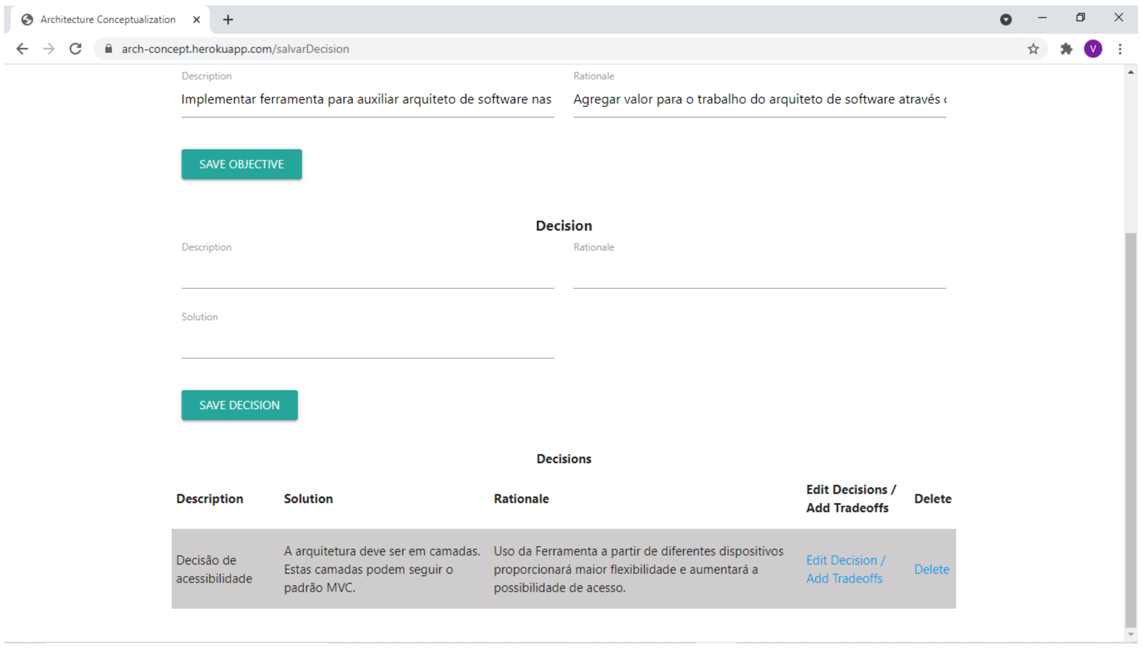
Description	Rationale
Decisão de acessibilidade	Uso da Ferramenta a partir de diferentes dispositivos proporcio

Solution: A arquitetura deve ser em camadas. Estas camadas podem segu

SAVE DECISION

Decisions

Description	Solution	Rationale	Edit Decisions / Add Tradeoffs	Delete


Architecture Conceptualization
arch-concept.herokuapp.com/salvarDecision
SAVE OBJECTIVE
SAVE DECISION

Description: Implementar ferramenta para auxiliar arquiteto de software nas

Rationale: Agregar valor para o trabalho do arquiteto de software através

Decision

Description

Rationale

Solution

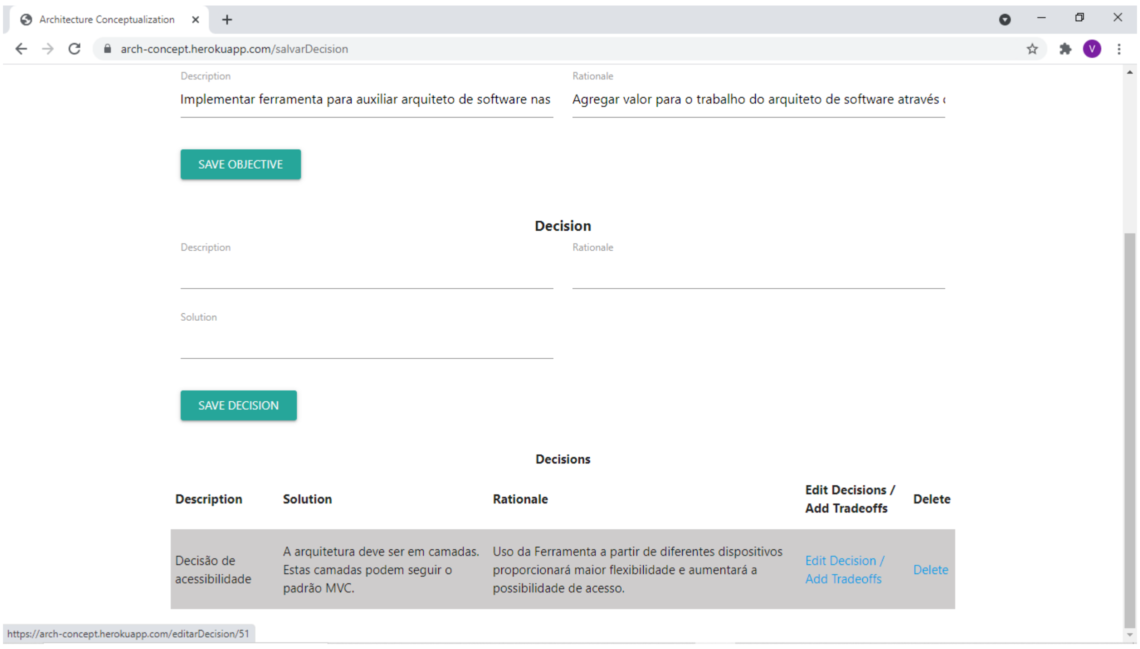
Decisions

Description	Solution	Rationale	Edit Decisions / Add Tradeoffs	Delete
Decisão de acessibilidade	A arquitetura deve ser em camadas. Estas camadas podem seguir o padrão MVC.	Uso da Ferramenta a partir de diferentes dispositivos proporcionará maior flexibilidade e aumentará a possibilidade de acesso.	Edit Decision / Add Tradeoffs	Delete

3.8 - Tela de Tradeoff

Clicar em “Add Tradeoffs” na linha correspondente a *Decision* que se deseja adicionar *Tradeoffs*.

Informar Tipo, Descrição e Racional do *Tradeoff*. Clicar em Save. Os dados cadastrados serão exibidos em tabela na parte inferior da tela.


Architecture Conceptualization

arch-concept.herokuapp.com/salvarDecision

Description: Implementar ferramenta para auxiliar arquiteto de software nas

Rationale: Agregar valor para o trabalho do arquiteto de software através

SAVE OBJECTIVE

Decision

Description:

Rationale:

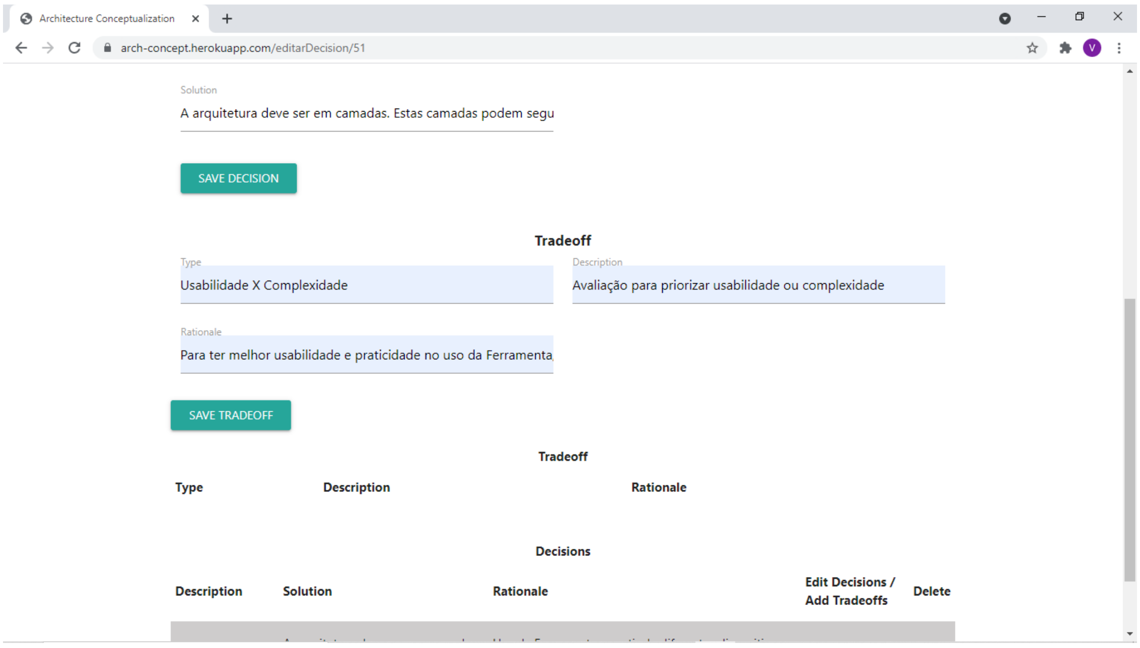
Solution:

SAVE DECISION

Decisions

Description	Solution	Rationale	Edit Decisions / Add Tradeoffs	Delete
Decisão de acessibilidade	A arquitetura deve ser em camadas. Estas camadas podem seguir o padrão MVC.	Uso da Ferramenta a partir de diferentes dispositivos proporcionará maior flexibilidade e aumentará a possibilidade de acesso.	Edit Decision / Add Tradeoffs	Delete

https://arch-concept.herokuapp.com/editarDecision/51


Architecture Conceptualization

arch-concept.herokuapp.com/editarDecision/51

Solution: A arquitetura deve ser em camadas. Estas camadas podem segu

SAVE DECISION

Tradeoff

Type: Usabilidade X Complexidade

Description: Avaliação para priorizar usabilidade ou complexidade

Rationale: Para ter melhor usabilidade e praticidade no uso da Ferramenta

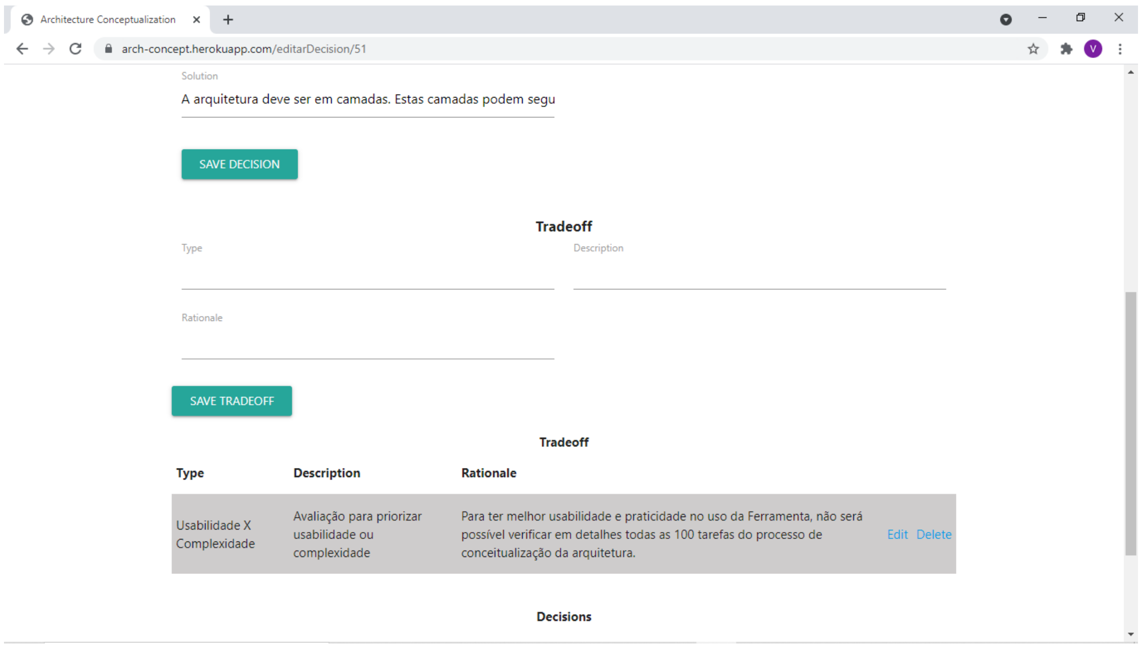
SAVE TRADEOFF

Tradeoff

Type	Description	Rationale

Decisions

Description	Solution	Rationale	Edit Decisions / Add Tradeoffs	Delete



Architecture Conceptualization

arch-concept.herokuapp.com/editarDecision/51

Solution

A arquitetura deve ser em camadas. Estas camadas podem segu

SAVE DECISION

Tradeoff

Type

Description

Rationale

SAVE TRADEOFF

Tradeoff

Type	Description	Rationale	
Usabilidade X Complexidade	Avaliação para priorizar usabilidade ou complexidade	Para ter melhor usabilidade e praticidade no uso da Ferramenta, não será possível verificar em detalhes todas as 100 tarefas do processo de conceitualização da arquitetura.	Edit Delete

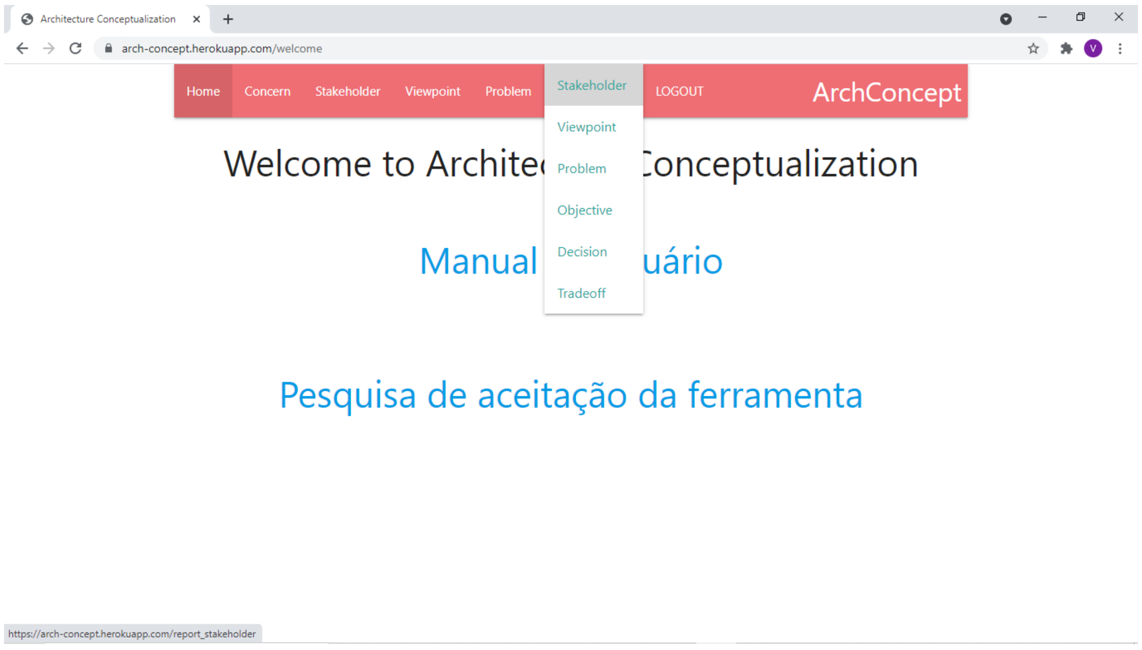
Decisions

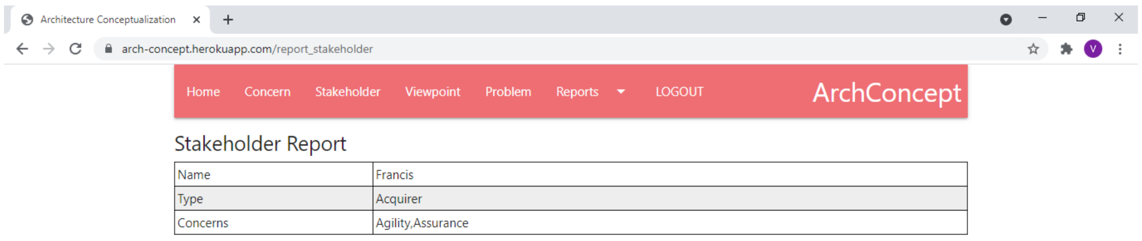
3.9 - Relatórios

São exibidos relatórios com os dados cadastrados, a partir do ponto de vista do Stakeholder.

Pode ser impresso ou gerado em arquivo *pdf* a partir do próprio navegador, visto que o relatório está formatado para impressão.

Stakeholders Report

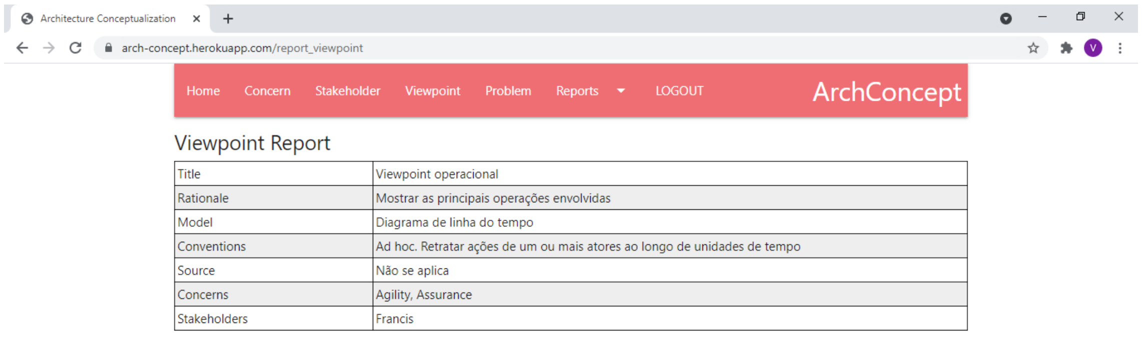

Architecture Conceptualization
arch-concept.herokuapp.com/welcome
Home
Concern
Stakeholder
Viewpoint
Problem
Stakeholder
Viewpoint
Problem
Objective
Decision
Tradeoff
LOGOUT
ArchConcept
Welcome to Archite Conceptualization
Manual uário
Pesquisa de aceitação da ferramenta
https://arch-concept.herokuapp.com/report_stakeholder


Architecture Conceptualization
arch-concept.herokuapp.com/report_stakeholder
Home
Concern
Stakeholder
Viewpoint
Problem
Reports
LOGOUT
ArchConcept

Stakeholder Report

Name	Francis
Type	Acquirer
Concerns	Agility,Assurance

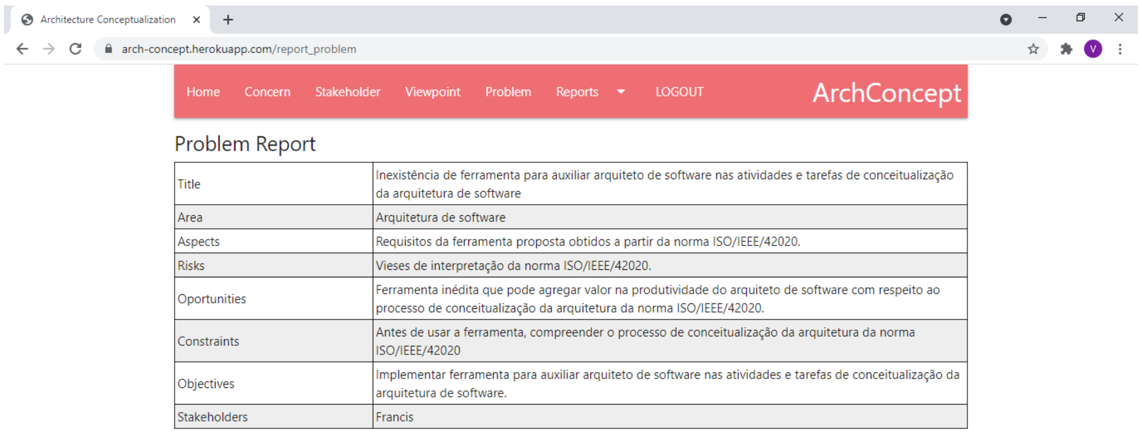
Viewpoint Report


Architecture Conceptualization
arch-concept.herokuapp.com/report_viewpoint
Home
Concern
Stakeholder
Viewpoint
Problem
Reports
LOGOUT
ArchConcept

Viewpoint Report

Title	Viewpoint operacional
Rationale	Mostrar as principais operações envolvidas
Model	Diagrama de linha do tempo
Conventions	Ad hoc. Retratar ações de um ou mais atores ao longo de unidades de tempo
Source	Não se aplica
Concerns	Agility, Assurance
Stakeholders	Francis

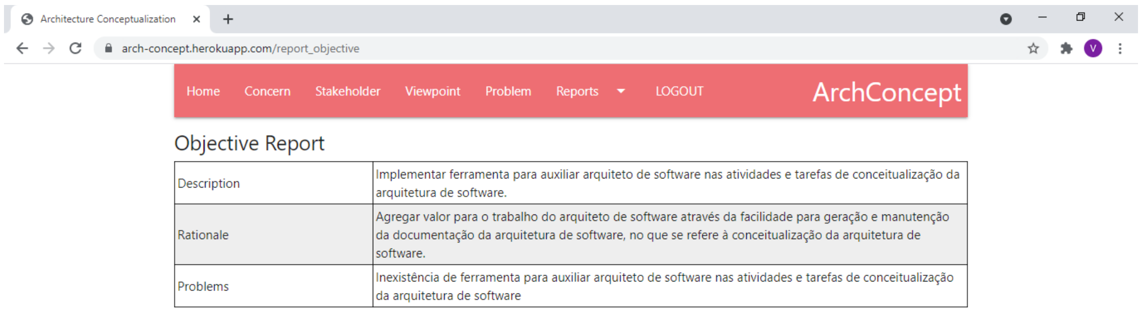
Problem Report


Architecture Conceptualization
arch-concept.herokuapp.com/report_problem
Home
Concern
Stakeholder
Viewpoint
Problem
Reports
LOGOUT
ArchConcept

Problem Report

Title	Inexistência de ferramenta para auxiliar arquiteto de software nas atividades e tarefas de conceitualização da arquitetura de software
Area	Arquitetura de software
Aspects	Requisitos da ferramenta proposta obtidos a partir da norma ISO/IEEE/42020.
Risks	Vieses de interpretação da norma ISO/IEEE/42020.
Oportunities	Ferramenta inédita que pode agregar valor na produtividade do arquiteto de software com respeito ao processo de conceitualização da arquitetura da norma ISO/IEEE/42020.
Constraints	Antes de usar a ferramenta, compreender o processo de conceitualização da arquitetura da norma ISO/IEEE/42020
Objectives	Implementar ferramenta para auxiliar arquiteto de software nas atividades e tarefas de conceitualização da arquitetura de software.
Stakeholders	Francis

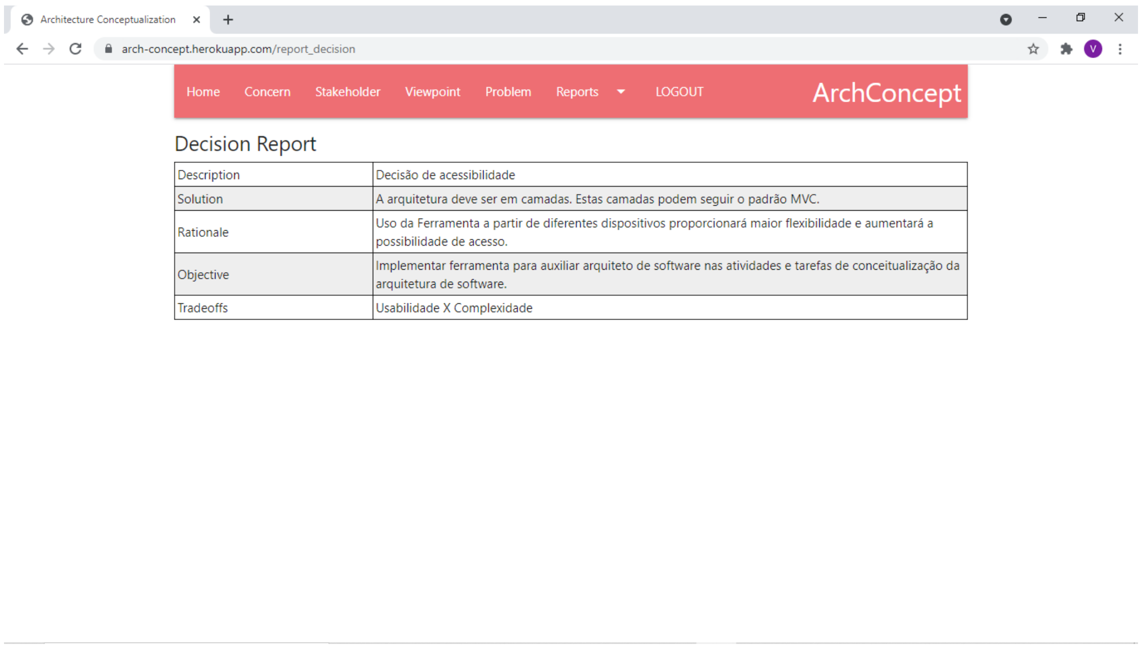
Objective Report


Architecture Conceptualization
arch-concept.herokuapp.com/report_objective
Home
Concern
Stakeholder
Viewpoint
Problem
Reports
LOGOUT
ArchConcept

Objective Report

Description	Implementar ferramenta para auxiliar arquiteto de software nas atividades e tarefas de conceitualização da arquitetura de software.
Rationale	Agregar valor para o trabalho do arquiteto de software através da facilidade para geração e manutenção da documentação da arquitetura de software, no que se refere à conceitualização da arquitetura de software.
Problems	Inexistência de ferramenta para auxiliar arquiteto de software nas atividades e tarefas de conceitualização da arquitetura de software

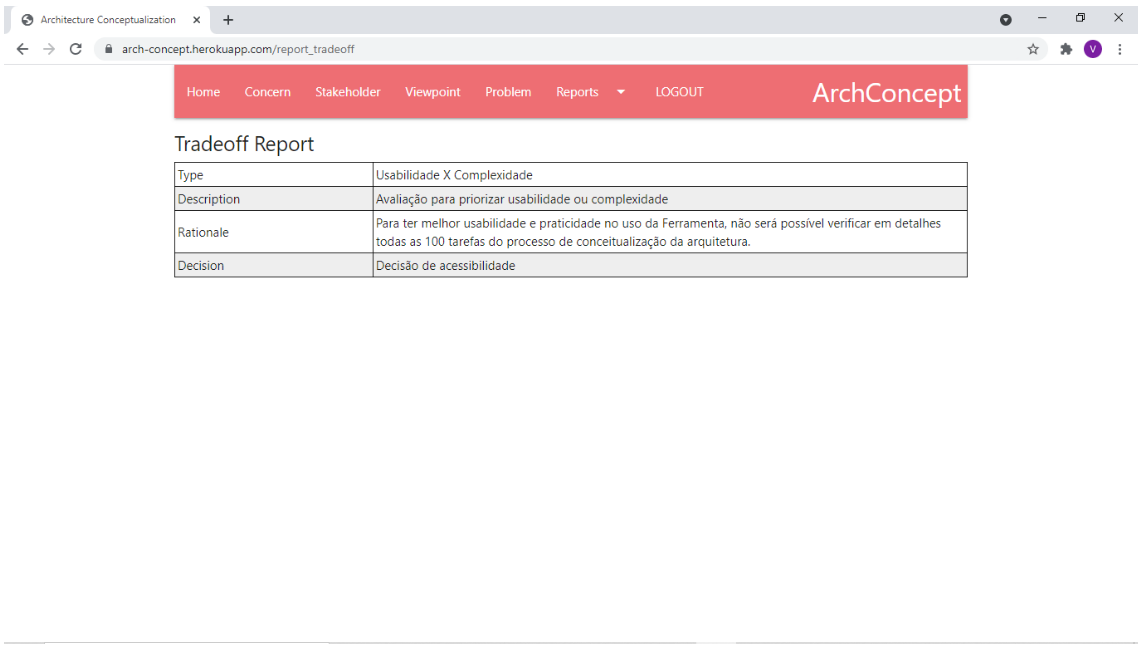
Decision Report


Architecture Conceptualization
arch-concept.herokuapp.com/report_decision
Home
Concern
Stakeholder
Viewpoint
Problem
Reports
LOGOUT
ArchConcept

Decision Report

Description	Decisão de acessibilidade
Solution	A arquitetura deve ser em camadas. Estas camadas podem seguir o padrão MVC.
Rationale	Uso da Ferramenta a partir de diferentes dispositivos proporcionará maior flexibilidade e aumentará a possibilidade de acesso.
Objective	Implementar ferramenta para auxiliar arquiteto de software nas atividades e tarefas de conceitualização da arquitetura de software.
Tradeoffs	Usabilidade X Complexidade

Tradeoff Report


Architecture Conceptualization
arch-concept.herokuapp.com/report_tradeoff
Home
Concern
Stakeholder
Viewpoint
Problem
Reports
LOGOUT
ArchConcept

Tradeoff Report

Type	Usabilidade X Complexidade
Description	Avaliação para priorizar usabilidade ou complexidade
Rationale	Para ter melhor usabilidade e praticidade no uso da Ferramenta, não será possível verificar em detalhes todas as 100 tarefas do processo de conceitualização da arquitetura.
Decision	Decisão de acessibilidade

4 - Exemplo de dados cadastrados

Conceitualização da Arquitetura da ferramenta "ArchConcept"

Stakeholder

Registro 1:

Name: André; Type: User.

Concern

Registro 1:

Description: Functionality.

Registro 2:

Description: Usage.

Viewpoint

Title: Viewpoint operacional

Rationale: Mostrar as principais operações envolvidas

Model: Diagrama de linha do tempo

Conventions: Ad hoc. Retratar ações de um ou mais atores ao longo de unidades de tempo

Source: Não se aplica

Concern: Functionality, usage, behavior

Problem

Title: Inexistência de ferramenta para auxiliar arquiteto de software nas atividades e tarefas de conceitualização da arquitetura de software

Area: Arquitetura de software

Aspects: Requisitos da ferramenta proposta obtidos a partir da norma ISO/IEEE/42020.

Risks: Vieses de interpretação da norma ISO/IEEE/42020.

Opportunities: Ferramenta inédita que pode agregar valor na produtividade do arquiteto de software com respeito ao processo de conceitualização da arquitetura da norma ISO/IEEE/42020.

Constraints: Antes de usar a ferramenta, compreender o processo de conceitualização da arquitetura da norma ISO/IEEE/42020

Stakeholder: André

Objective

Description: Implementar ferramenta para auxiliar arquiteto de software nas atividades e tarefas de conceitualização da arquitetura de software.

Rationale: Agregar valor para o trabalho do arquiteto de software através da facilidade para geração e manutenção da documentação da arquitetura de software, no que se refere à conceitualização da arquitetura de software.

Decision

Description: Decisão de acessibilidade

Solution: A arquitetura deve ser em camadas. Estas camadas podem seguir o padrão MVC.

Rationale: Uso da Ferramenta a partir de diferentes dispositivos proporcionará maior flexibilidade e aumentará a possibilidade de acesso.

Tradeoff

Type: Usabilidade X Complexidade

Description: Avaliação para priorizar usabilidade ou complexidade

Rationale: para ter melhor usabilidade e praticidade no uso da Ferramenta, não será possível verificar em detalhes todas as 100 tarefas do processo de conceitualização da arquitetura.

APPENDIX B – TAM Questionnaire

Questionário de análise de aceitação da ArchConcept, uma ferramenta para conceitualização da Arquitetura de Software*

Prezado(a) participante:

Esta pesquisa faz parte de Dissertação de Mestrado do Programa de Pós-Graduação em Ciência da Computação, linha de pesquisa Engenharia de Software, Fundação Universidade Federal de Sergipe. A pesquisa está sob supervisão do professor Dr. Michel dos Santos Soares.

O objetivo da pesquisa é avaliar a aceitação da ArchConcept, uma ferramenta experimental baseada no Processo de Conceitualização da Arquitetura de Software, integrante da Norma ISO/IEC/IEEE 42020.

Sua participação envolve a resposta de um questionário que tem a duração aproximada de 10 minutos.

A participação nesse estudo é voluntária e se você decidir não participar ou quiser desistir
de continuar em qualquer momento, tem absoluta liberdade de fazê-lo. Na publicação dos
resultados desta pesquisa, sua identidade será mantida no mais rigoroso sigilo. Serão omitidas todas as informações que permitam identificá-lo(a).

Para esta análise será utilizado o modelo TAM (Technology Acceptance Model), que se baseia em 3 características: Utilidade Percebida (Perceived Usefulness – PU), Facilidade
de Uso Percebida (Perceived Ease of Use - PEOU) e Uso Percebido (Usage Perceived - UP).
As características citadas serão detalhadas a cada seção de questões.

Quaisquer dúvidas relativas à pesquisa poderão ser esclarecidas pelo pesquisador - Valdicélio Mendes (Mestrando), pelo email vmsantos@dcomp.ufs.br

*Pedimos responder este questionário imediatamente após utilizar a ArchConcept.

OBS.: Gostaríamos de pedir a sua compreensão e cooperação para que os dados coletados sejam consistentes. Pedimos que cada usuário preencha o questionário apenas
uma vez.

Legenda:

1- Discordo Totalmente
2- Discordo
3- Neutro
4- Concordo
5- Concordo Totalmente

*Obrigatório

1. Qual seu nome? (Opcional)

____

2. Qual seu cargo atual? *

Marcar apenas uma oval.

- ◯ Analista de Sistemas
- ◯ Arquiteto de Software/Sistemas
- ◯ Líder de Sistemas
- ◯ Gerente de Projetos
- ◯ Professor Universitário
- ◯ Pesquisador/Estudante de Pós-Graduação
- ◯ Outro: ____________________

3. Qual a sua formação em nível de graduação? *

Marcar apenas uma oval.

- ◯ Ciência da Computação
- ◯ Sistemas de Informação
- ◯ Engenharia da Computação
- ◯ Engenharia de Software
- ◯ Engenharia Elétrica
- ◯ Curso Tecnológico
- ◯ Outro: ____________________

4. Qual seu grau máximo de formação? *

Marcar apenas uma oval.

- Graduação
- Especialização
- MBA
- Mestrado
- Doutorado

5. Tempo de experiência em Desenvolvimento de Software (em anos) *

6. Tempo de experiência em Arquitetura de Software (em anos) *

7. Já utilizou alguma ferramenta para conceitualização da Arquitetura de Software? *

Marcar apenas uma oval.

- Sim
- Não

8. Se Sim, qual?

9. I - Utilidade Percebida (Perceived Usefulness – PU). Característica que verifica em que grau uma pessoa acredita que ao usar um sistema específico elevaria seu desempenho no trabalho. Ou seja, quais funções estão presentes na ferramenta para melhorar o relacionamento uso-desempenho do usuário. *

Marcar apenas uma oval por linha.

	1	2	3	4	5
Usar a ArchConcept possibilita melhor gerenciamento das minhas atividades no trabalho.	○	○	○	○	○
2. Usar a ArchConcept aumenta meu desempenho no trabalho.	○	○	○	○	○
3. Na ArchConcept, as funcionalidades de cadastramento, edição e exclusão de stakeholders são úteis ao meu trabalho.	○	○	○	○	○
4. Na ArchConcept, as funcionalidades de inclusão e exclusão de concerns são úteis ao meu trabalho.	○	○	○	○	○
5. Na ArchConcept, as funcionalidades	○	○	○	○	○

de cadastramento, edição e exclusão de viewpoints são úteis ao meu trabalho.					
6. Na ArchConcept, as funcionalidades de cadastramento, edição e exclusão de problems são úteis ao meu trabalho.	○	○	○	○	○
7. Na ArchConcept, as funcionalidades de cadastramento, edição e exclusão de objectives são úteis ao meu trabalho.	○	○	○	○	○
8. Na ArchConcept, as funcionalidades de cadastramento, edição e exclusão de decisions são úteis ao meu trabalho.	○	○	○	○	○
9. Na ArchConcept, as funcionalidades de cadastramento, edição e	○	○	○	○	○

exclusão de tradeoffs são úteis ao meu trabalho.					
10. A ArchConcept possibilita executar minhas tarefas em menos tempo.	○	○	○	○	○
11. Usar a ArchConcept aumenta minha produtividade.	○	○	○	○	○

10. II - Facilidade de Uso Percebida (Perceived Ease of Use - PEOU). Característica que se refere ao grau para o qual a pessoa acredita que ao utilizar um sistema específico, este o deixaria livre de esforços. Assim, nesse quesito iremos analisar que facilidades a ferramenta oferece. *

Marcar apenas uma oval por linha.

	1	2	3	4	5
1. Para mim, é fácil cadastrar, editar ou excluir stakeholders.	◯	◯	◯	◯	◯
2. Para mim, é fácil correlacionar stakeholders e concerns.	◯	◯	◯	◯	◯
3. Para mim, é fácil cadastrar, editar ou excluir viewpoints.	◯	◯	◯	◯	◯
4. Para mim, é fácil cadastrar um viewpoint a partir do concern já cadastrado.	◯	◯	◯	◯	◯
5. Para mim, é fácil cadastrar, editar ou excluir problems.	◯	◯	◯	◯	◯
6. Para mim, é fácil cadastrar um problem, associando-o a um ou mais stakeholders.	◯	◯	◯	◯	◯
7. Para mim, é fácil cadastrar, editar ou excluir objectives.	◯	◯	◯	◯	◯
8. Para mim, é	◯	◯	◯	◯	◯

8. Para mim, é fácil cadastrar um ou mais objectives a partir de um problem já cadastrado.					
9. Para mim, é fácil cadastrar, editar ou excluir decisions.	○	○	○	○	○
10. Para mim, é fácil cadastrar uma ou mais decisions a partir de um objective previamente cadastrado.	○	○	○	○	○
11. Para mim, é fácil cadastrar, editar ou excluir tradeoffs.	○	○	○	○	○
12. Para mim, é fácil cadastrar um ou mais tradeoffs a partir de uma decision previamente cadastrada.	○	○	○	○	○
13. Para mim, é fácil gerar relatórios de conceitualização da arquitetura.	○	○	○	○	○
14. É fácil se recordar de como executar as tarefas na ArchConcept.	○	○	○	○	○
15. Consigo utilizar a ArchConcept	○	○	○	○	○

sem consultar o manual do usuário.					
16. Usar a ArchConcept torna meu trabalho mais fácil de realizar.	◯	◯	◯	◯	◯
17. Foi fácil para mim utilizar a	◯	◯	◯	◯	◯

11. III - Uso Percebido (Usage Perceived - UP). Característica que avalia o uso efetivo da ferramenta ArchConcept. Que funções realmente são utilizadas. *

Marcar apenas uma oval por linha.

	1	2	3	4	5
1. Estou interessado em conhecer mais o processo de Conceitualização da Arquitetura, presente na Norma ISO/IEC/IEEE 42020.	◯	◯	◯	◯	◯
2. Os relatórios gerados pela ArchConcept são úteis para mim.	◯	◯	◯	◯	◯
3. A ArchConcept é útil ao meu trabalho.	◯	◯	◯	◯	◯
4. Utilizaria pelo menos uma das funcionalidades da ArchConcept em meu trabalho.	◯	◯	◯	◯	◯
5. Utilizaria a maior parte ou todas as funcionalidades da ArchConcept em meu trabalho.	◯	◯	◯	◯	◯

12. Caso queira fazer um comentário sobre a ferramenta, o questionário ou sobre o processo de Conceitualização da Arquitetura de Software, escreva a seguir: