



ALEXANDRA SERRA LÉLÉ

BSc in Computer Science Engineering

AN APPROACH FOR HUMAN VALUES REQUIREMENTS: THE CASE OF UNIVERSALISM

MASTER IN COMPUTER SCIENCE AND ENGINEERING

NOVA University Lisbon
September, 2024

AN APPROACH FOR HUMAN VALUES REQUIREMENTS: THE CASE OF UNIVERSALISM

ALEXANDRA SERRA LÉLÉ

BSc in Computer Science Engineering

Adviser: João Baptista da Silva Araújo Junior

Associate Professor, NOVA University Lisbon

Co-adviser: Ana Maria Diniz Moreira

Full Professor, NOVA University Lisbon

Examination Committee

Chair: Dr. Nuno Manuel Ribeiro Preguiça

Full Professor, NOVA University Lisbon

Rapporteur: Dr. Carla Taciana Lima Lourenço Silva Schuenemann

Associate Professor, Universidade Federal de Pernambuco

Member: Dr. João Baptista da Silva Araújo Júnior

Associate Professor, NOVA University Lisbon

An Approach for Human Values Requirements: The Case of Universalism

Copyright © Alexandra Serra Lélé, NOVA School of Science and Technology, NOVA University Lisbon.

The NOVA School of Science and Technology and the NOVA University Lisbon have the right, perpetual and without geographical boundaries, to file and publish this dissertation through printed copies reproduced on paper or on digital form, or by any other means known or that may be invented, and to disseminate through scientific repositories and admit its copying and distribution for non-commercial, educational or research purposes, as long as credit is given to the author and editor.

ACKNOWLEDGEMENTS

I want to thank my advisors, João Araújo and Ana Moreira, for their guidance. Their availability and insights have been instrumental in completing this dissertation.

To my family, especially my parents, thank you for the love, encouragement, and support that made it possible for me to pursue and finish my studies and this dissertation.

I can't forget my friends whose support and companionship throughout this journey meant everything to me. Thank you, everyone, for being by my side. I wish you all the very best in life.

ABSTRACT

Human values differ in importance for each individual, influencing behaviors and defining priorities. These values shape people's actions and dictate where they allocate their attention. Even knowing their strong influence on humanity, software systems often fail to accommodate all recognized human values. Some values receive greater focus and preference over others. It's possible to assert this because specific values, such as security, are widely examined and prioritized. In contrast, values like universalism, which includes aspects such as equality, social justice, and wisdom, tend to receive relatively less attention.

For this reason, this dissertation aims to provide a requirements engineering approach for human values, focusing on one value in particular: Universalism. This value's goal is to promote understanding and tolerance among everyone, safeguarding the welfare of all people and nature, explaining why it should not be ignored, and why it is essential to integrate it into software development.

The primary step of this dissertation was a systematic mapping study to explore what currently exists in software engineering that supports the integration of Universalism values. This step is the foundation of a conceptual model for universalism requirements. This model provides requirements engineers with an understanding of universalism requirements during the requirements elicitation phase. Then, a framework was proposed to integrate the model into existing practices of requirements engineers from the initial phase and throughout the development process. Next, a supporting tool, a prototype, was developed to facilitate its use and integration. The model was evaluated by a group of 24 participants, and the results were promising.

Keywords: Requirements Engineering, Human Values, Universalism

RESUMO

Os valores humanos diferem em importância para cada indivíduo, influenciando comportamentos e definindo prioridades. Esses valores moldam as ações das pessoas e ditam onde elas colocam a sua atenção. Mesmo sabendo da sua forte influência sobre a humanidade, os sistemas de software muitas vezes não conseguem acomodar todos os valores humanos reconhecidos. Alguns valores recebem maior foco e preferência do que outros. É possível afirmar isto porque valores específicos, como a segurança, são amplamente examinados e priorizados. Em contraste, valores como o universalismo, que inclui aspectos como igualdade, justiça social e sabedoria, tendem a receber relativamente menos atenção.

Por esta razão, esta dissertação pretende propor uma abordagem de engenharia de requisitos para valores humanos, focando-se num valor em particular: *Universalism*. O objetivo deste valor é promover a compreensão e a tolerância entre todos, salvaguardando o bem-estar de todas as pessoas e da natureza, explicando porque não deve ser ignorado e porque é essencial integrá-lo no desenvolvimento de software.

O primeiro passo desta dissertação foi um estudo de mapeamento sistemático para explorar o que existe atualmente em engenharia de software que apoia a integração de valores de *Universalism*. Esta etapa é, portanto, a base para o desenvolvimento de um modelo conceptual para requisitos de *Universalism*. Este modelo fornece uma compreensão dos requisitos de *Universalism* durante a fase de elicitação de requisitos. De seguida, foi proposta uma framework para integrar o modelo nas práticas existentes da engenharia de requisitos desde a fase inicial e durante o processo de desenvolvimento. Depois foi desenvolvida uma ferramenta de apoio, um protótipo, para facilitar a sua utilização e integração. O modelo foi avaliado por um grupo de 24 participantes e os resultados foram promissores.

Palavras-chave: Engenharia de Requisitos, Valores Humanos, *Universalism*

CONTENTS

List of Figures	viii
List of Tables	x
1 Introduction	1
1.1 Context and Motivation	1
1.2 Problem statement	3
1.3 Objectives	4
1.4 Contributions	4
1.5 Structure of the document	5
2 Background	6
2.1 Requirements engineering	6
2.2 Human Values	11
2.2.1 Relations between values	12
2.2.2 Factors that influence values	13
2.2.3 Universalism value	15
2.3 Systematic mapping studies	17
2.4 Summary	19
3 Related Work	21
3.1 Search Process	21
3.1.1 Planning Phase	21
3.1.2 Conducting Phase	26
3.1.3 Analysis and Mapping Phases	28
3.2 Discussion	29
3.3 Threats to validity	29
3.4 Summary	30
4 Conceptual Modeling of Universalism Requirements	31

4.1	Building the conceptual model	31
4.1.1	Specification	31
4.1.2	Knowledge Acquisition	32
4.1.3	Conceptualization	33
4.2	A conceptual model for Universalism requirements	33
4.2.1	Equality	35
4.2.2	Security and Privacy	41
4.2.3	Culture	44
4.2.4	Gender	47
4.2.5	Human	50
4.2.6	Software System	52
4.3	Summary	52
5	A Framework for Eliciting Universalism Requirements	54
5.1	Framework foundations	54
5.2	A process to use the Framework	56
5.2.1	Pre-elicitation	56
5.2.2	Plan the elicitation	57
5.2.3	Elicitation process	57
5.3	A Supporting Tool for the Framework	57
5.3.1	Prototype development	58
5.4	Applying the Framework to a Case Study	63
5.5	Summary	67
6	Evaluation	68
6.1	Evaluation planning	68
6.1.1	Questionnaire	68
6.1.2	Prototype Guides	70
6.1.3	Framework Document	70
6.2	Results and discussion	70
6.2.1	Participants demographics	70
6.2.2	Components	73
6.2.3	Conceptual model	79
6.2.4	Prototype Evaluation	85
6.3	Threats to validity	94
6.3.1	Internal Validity	94
6.3.2	External Validity	94
6.3.3	Construct Validity	95
6.3.4	Conclusion Validity	95
6.4	Summary	95
7	Conclusions and Future Work	97

7.1 Contributions	97
7.2 Future work	98
Bibliography	100
Appendices	
A Evaluation Questionnaire	110

LIST OF FIGURES

2.1	Ten Steps Requirements Elicitation Process from [89].	9
2.2	Schwartz's theory of basic values from [34].	12
2.3	Theoretical model of relations among ten motivational types of values, adapted from [86].	14
4.1	Conceptual model of universalism requirements.	34
4.2	Equality part of the conceptual model.	35
4.3	Security and Privacy part of the conceptual model.	41
4.4	Culture part of the conceptual model.	44
4.5	Gender part of the conceptual model.	47
5.1	Landing page of Universalism tool.	58
5.2	Concept selection from the model.	59
5.3	Characteristics of the selected concept.	59
5.4	Questions' selection of the selected concepts.	60
5.5	ShareTheMeal app description.	60
5.6	Document page with the form for the details.	61
5.7	Documents page with the list of documents.	61
5.8	The final requirements document.	62
5.9	The Freerice requirements document.	66
6.1	Percentage of participants by gender.	71
6.2	Percentage of participants by age.	71
6.3	Percentage of participants by academic training.	71
6.4	Percentage of participants by scientific area of the degree.	72
6.5	Percentage of participants by current occupation.	72
6.6	Percentage of participants by years of experience.	72
6.7	In your opinion, how important are universalism values - such as equality, social justice - during software development?	73

6.8	Does this model properly include all the necessary concepts and relations regarding Security and Privacy?	74
6.9	Does this model properly include all the necessary concepts and relations regarding Equality?	76
6.10	Does this model properly include all the necessary concepts and relations regarding Culture?	77
6.11	Does this model properly include all the necessary concepts and relations regarding Gender?	78
6.12	Does this model properly include all the necessary concepts and relations regarding Software System?	78
6.13	Does this model properly include all the necessary concepts and relations regarding Human?	79
6.14	How useful is this conceptual model?	80
6.15	How difficult is it to understand this conceptual model?	81
6.16	Would you use this conceptual model to develop an application where universalism requirements are relevant?	81
6.17	Would you recommend this conceptual model?	82
6.18	About the tool page, did you have any difficulty in exploring the model? . .	86
6.19	About the tool page, did you have any difficulty in the selection and saving of the concepts?	87
6.20	About the questions page, did you have any difficulty in selecting and removing the concepts?	88
6.21	Was the case example of the app ShareTheMeal useful?	88
6.22	About the document page, did you have any difficulty in adding and removing participants?	89
6.23	About the documents page, did you have any difficulty in exporting, editing or deleting a document?	90
6.24	How intuitive would you rate the interface?	90
6.25	How is the navigation of the tool?	90
6.26	Was the document useful for your understanding of the model?	92
6.27	In general, was the description of the concepts comprehensible?	92
6.28	Was the guide useful?	93
6.29	Was the guide easy to understand?	93
6.30	Was the video useful?	93

LIST OF TABLES

2.1	Schwartz Values, adapted from [86].	13
3.1	PICOC Analysis.	22
3.2	Research Sub-questions.	22
3.3	Search Keywords.	23
3.4	Inclusion and Exclusion Criteria for the first iteration.	25
3.5	Inclusion and Exclusion Criteria for the second iteration.	25
3.6	Data extracted into the worksheet.	26
3.7	Results of the automatic search.	27
3.8	Titles of the 10 papers selected in the automated and manual search.	27
5.1	Proposed framework phases.	56

INTRODUCTION

The present study stems from the need to consider human values when developing software. Every person has values (for example, safety, responsibility, ambition, honesty) that might differentiate them from others and vary in their level of importance. Software systems must conform to human values as much as possible to satisfy users' necessities, and failure to do so leads to negative socioeconomic impacts. The goal of this dissertation is to design a conceptual model that integrates human values requirements in software development, specifically the integration of Universalism values. Why concentrate on Universalism?

Several values have already been vastly explored, such as Security, while others are overlooked, like tradition, power, conformity, and universalism. The latter's purpose is understanding, appreciation, tolerance, and protection for the welfare of all people and nature. For this reason, Universalism is essential in software systems since it ensures inclusivity and fairness, promoting compatibility and accessibility for diverse users.

Consequently, the decision was to investigate this specific value type over the rest.

This initial chapter provides an overview of the context and motivation of this dissertation, describes the problem statement and the associated challenges, and explains the objectives of this dissertation along with the expected contributions.

1.1 Context and Motivation

At the heart of human behavior resides values. These values are criteria that dictate the way people choose to judge actions, people, and events. Different people have different values, and, as a result, their choices and reactions towards everything that surrounds them in the world, such as religion, gender, culture, and others, differ from person to person. A person's upbringing, experience, and knowledge impact the core values as someone evolves. Among numerous values, examples include respect, tolerance, safety, and pleasure, among countless others [85]. Ignoring these values in software causes user dissatisfaction and has a negative consequence in relation to socioeconomics, for instance, financial and credibility loss. The public presses those in the software business

to incorporate their values in software development to prevent violations of their values. This occurrence arises from the growing understanding of the weight of human factors in software [64]. One example is Google, which promised, in 2018, not to use AI as a weapon or surveillance instrument in a way that would violate norms or against human rights [14]. However, this promise seems compromised given recent actions concerning the company's involvement in Project Nimbus, a cloud computing contract with Israel that could be used as surveillance, violating human rights [28]. This contradiction showcases the challenges in aligning companies' practices with ethical principles, and thus, demonstrates the importance of integrating human values in software like transparency and accountability.

Despite the many attempts to integrate human values in software, a sound balance has yet to be achieved [54]. This absence of equilibrium refers to the limited consideration of the countless values during system development. Sometimes, achieving complete success as an outcome is not guaranteed.

In the late 20th century, overlooked key features in software systems designs gained momentum. These features include usability, interface, and even safety [20].

As reported by [54], the failure to integrate human values in software projects occurs for three reasons. The first is the shortage of practical definitions for human values in software engineering. A value may imply different meanings depending on the context. For example, security in software engineering represents confidentiality and integrity, but in social sciences, this may signify a sense of belonging. So, a practical definition is essential in software engineering practices [54]. The second reason is the disregard of values in software design decisions. As said previously, numerous values are neglected in software design, while others, like privacy, receive vast attention and study. Software practitioners haven't globally adopted a solution integrated into software design tools [54]. The final reason is the lack of measurement of human values in software. The measuring of these is crucial for the evaluation of decisions concerning human values. Some metrics exist, but only for some values. A significant number of them, don't have such metrics, including social justice, gender equality, and respect for tradition, among others [54]. Why does this happen? The answer might be the non-existence of attention in the development of practical definitions for human values in software engineering practice [54].

Before making any step in conceptualizing Human Values in software engineering, it's necessary to comprehend the definitions of Human Values. A highly debated and widely discussed concept is the Theory of Universal Values developed by Schwartz, which defines ten distinct basic values characterized by diverse goals and motivations. Universalism is among these values, it stands for the understanding, tolerance, and protection of the well-being of all people and nature. Furthermore, these values can be divided into sub-values [86].

The growing awareness of the missing values in software is an issue that needs a solution for a better future and advancement in the field of software development. Security is not the only important aspect of systems, other features are essential for the society of today and tomorrow. To contribute to the improvement of software, this dissertation exists

for values not explored as extensively, in this case, Universalism, in software engineering.

1.2 Problem statement

The problem statement of this dissertation is the *lack of software development approaches that integrate universalism values*. The discussion that ensues elaborates on the identified issue and the consequent challenges that derive from it.

Although more methodologies and frameworks have emerged in recent years to address the missing values from software systems, such as gender-inclusive, cultural aspects, sustainability, and so on, they are still limited when compared to a value like security. Due to the lack of approaches, the negative impact of ignoring human values during software development causes user dissatisfaction and negative consequences in socioeconomics, for example, financial and credibility loss [64]. A value so essential as universalism, which has a lot of important sub-values like, to name a few, equality, social justice, and environmental [86], needs to be widely considered in software development, mainly requirements engineering. Addressing these values from the beginning assists in gathering essential requirements for a system that satisfies everyone. There are several challenges that originate from this problem, which will be explored below.

Challenge 1: Characterize the state of the art

Before addressing any challenge, the first task of this dissertation is to effectuate a deep analysis of the subject in question. This requires careful and thoughtful execution. An ad hoc search, where one merely searches in databases and search engines as needed with no predefined criteria, won't be enough. Instead, the approach selected was a systematic mapping study to explore how requirements engineering addresses the values of universalism. In later chapters, the results will be further discussed and related.

Challenge 2: Develop a definition and conceptualization of Universalism values

The second challenge is about understanding the development of a software system in such a way as to integrate the values of universalism. As such, it's necessary to initially comprehend the system's purpose and the requirements for the software to operate with those values in mind. For this, the following aspects are imperative: A definition and conceptualization of universalism requirements, the applicable contexts, and the information needed for their formulation. With all these in hand, everyone can understand and discuss the process to reach an agreement that satisfies all parties, obtaining a system following the values of universalism. So, for that end, the papers from the systematic mapping study were employed.

Challenge 3: Develop an approach for eliciting Universalism requirements

Knowing the definition and conceptualization of universalism requirements is not enough. It's hard to define a set of guidelines that apply to different contexts, especially in a value like universalism that accommodates a wide variety of differences that can be cultural, social, and individual. Assuring the equality of the requirements for these dimensions is not easy. There needs to exist a balance in the criteria in a way that diversity is acknowledged, but, at the same time, the standards must also be universal and able to be applied impartially.

For example, the design of the privacy settings in a social media platform could vary across countries based on the weight they attribute to privacy. In countries where privacy is highly valued, users might expect more options to control who can view their information. In others, users may prefer fewer settings. The approach in this case would need to respect these differences and, at the same time, have a standard that ensures all data is handled equally, regardless of their preferences.

1.3 Objectives

The main objective of this dissertation is to *propose a conceptual model for Universalism requirements and a framework to incorporate this model into requirements elicitation*. The priority will be understanding the values of universalism and their impacts on software to reach this goal. To effectively address universalism values, an evaluation of methodologies, requirements techniques, tools, models, and others will be performed.

1.4 Contributions

Via this dissertation, we contribute to the requirements engineering field with the following:

- A systematic mapping study on how universalism values have been addressed in requirements engineering;
- A conceptual model for universalism requirements based on the systematic mapping study;
- A framework for incorporating the conceptual model into requirements elicitation practices;
- A prototype tool designed to support the framework, assisting in the planning of requirements elicitation;
- An evaluation of the conceptual model, analyzing the participants' views on the usability and comprehension of the prototype tool.

1.5 Structure of the document

This document contains four chapters with the following structure:

- **Chapter 1:** *Introduction* introduces the context and motivation for this dissertation, determines the problem statement and the respective challenges, and finishes by explaining the objectives and expected contributions.
- **Chapter 2:** *Background* describes the concepts covered throughout this dissertation preparation, which consists of Requirements engineering, Human Values, and Systematic mapping studies.
- **Chapter 3:** *Related Work* demonstrates the systematic mapping study performed on the subject of this dissertation, aiming to answer the research question: How are Universalism and its sub-values addressed in the elicitation and specification activities in Requirements Engineering? The chapter further analyzes and discusses the obtained results.
- **Chapter 4:** *Conceptual Modeling of Universalism Requirements* presents a conceptual model for universalism requirements based on the systematic mapping study, describing the model development and foundations.
- **Chapter 5:** *A Framework for Eliciting Universalism Requirements* proposes a framework for integrating the conceptual model in requirements elicitation. It also describes the development of a universalism prototype tool through the process of exposing how to work with the framework.
- **Chapter 6:** *Evaluation* reports the conceptual model evaluation, the planning, and a discussion of the results.
- **Chapter 7:** *Conclusions and Future Work* concludes the dissertation and presents opportunities for future work.

BACKGROUND

This chapter aims to improve the understanding of essential concepts in this research work, such as Requirements Engineering, Human Values, and Systematic Mapping Studies. First, a description of requirements engineering will be given. Second, the concept of Human Values and its relevance in software development will be addressed. Finally, an explanation of the method employed in Chapter 3, systematic mapping studies, is provided.

2.1 Requirements engineering

Requirements engineering is defined as "the branch of software engineering concerned with the real-world goals for, functions of, and constraints on software systems. It is also concerned with the relationship of these factors to precise specifications of software behavior, and to their evolution over time and across software families" [107].

But what are requirements? They are traits belonging to a system that ensures its success in the applied domain [25]. Requirements are essential in software products, as such, requirements engineering is a crucial process in developing a system [84]. Functioning as the sector dedicated to tying the development activities to real-world challenges, it enables the analysis of the suitability and cost-effectiveness of the resulting solution. The activities of requirement engineering guarantee that the attributes that the stakeholders want in a system are documented in a manner that facilitates its analysis and ensuing implementation. All of this is done to ensure the system design is precise to meet the users' expectations. Clarifying the term stakeholders, these are individuals or organizations who depend on the success or failure of a system [58]. Requirements Engineering process has five main activities with the goal of helping comprehend a customer's needs, determining, analyzing, and evaluating the system requirements and constraints, and validating and managing the requirements specifications. These activities are [88]:

- **Requirements Elicitation:** This activity's purpose is to gather the requirements for the system that will be developed. This is the first phase in requirements engineering, and it's viewed as the process of determining the system requirements

and constraints according to the stakeholders' objectives and the users' needs. The exploration of possible alternatives is also an activity that falls in this scope.

- **Requirements Analysis and Negotiation:** This activity's goal is to learn of existing problems in the system requirements and solve them according to the stakeholders' necessities. Sometimes, requirements are either incomplete, conflicting, or ambiguous. The analysis phase reviews all requirements elicited to find any inconsistencies that would classify as a problem in the system. After these findings come the negotiation phase, where discussions ensue concerning the conflicts in requirements, and a compromise must be reached with the stakeholders. This entire process has to remain objective, based only on technicalities and organizational needs.
- **Requirements Documentation:** This activity aims to achieve the objective of documenting the requirements agreed upon in the second phase in a consistent way. After the previous phase analysis, the requirements are recorded formally with the appropriate specification mechanism. This phase is indispensable because it's the basis other development stages depend on, so the organization must transmit clarity and consistency.
- **Requirements Validation:** In this activity, the focus is on verifying the former work. Any conflicts, omissions, and inconsistencies in the requirements document go through corrections. The result is a document that represents precisely the stakeholders' needs.
- **Requirements Management:** As a software system evolves, requirements may change, so a necessary part of the maintenance task is an update of these requirements. These alterations are documented to keep up with the ongoing evolution of the system and maintain consistency.

These are the five fundamental activities in the requirements engineering process. The focus of this dissertation will be on requirements elicitation, detailed below.

Requirements Elicitation

Requirements Engineering is an activity "to discover, document, and maintain a set of requirements." However, it is not a single activity but a process with numerous phases and actions. One of them is the elicitation phase, the very first step in gathering user requirements [4].

Requirements elicitation, as was previously said, concentrates on learning and comprehending the needs of users and stakeholders to convey them to the developers. This process generally encloses five main types of standard activities [108]:

- **Understanding the Application Domain:** In the initial phase of requirements elicitation, it's necessary to investigate entirely the real-world context or application domain of the system in question, exploring the environment, comprising aspects

such as political, organizational, and social, and identifying constraints. Describing existing work processes and associated problems about essential business objectives and issues.

- **Identifying the Sources of Requirements:** Requirements derive from diverse sources and formats. Stakeholders, users, and subject matter experts provide insights into problems and user needs. Existing systems, processes, and documentation, particularly in projects substituting current systems, offer valuable details about the organization, environment, and supporting justifications for new system requirements.
- **Analyzing the Stakeholders:** Stakeholders are crucial in requirements elicitation, and they vary from internal and external groups to individuals, with the customer, especially the project sponsor, holding the principal role. In some instances, the actual users are the most necessary. Other groups, like those managing work processes and partners, should be viewed as stakeholders if affected. An initial step in requirements elicitation involves analyzing and involving all relevant stakeholders.
- **Selecting the Techniques, Approaches, and Tools to Use:** Some say that a single elicitation technique or methodology is enough, but it is normally recognized that one approach cannot accommodate all projects. The selection of elicitation techniques depends on the project, impacting the success of the process. Requirements elicitation functions best when using diverse techniques, often in collaboration and at different steps of the software development life cycle.
- **Eliciting the Requirements from Stakeholders and Other Sources:** After identifying requirement sources and stakeholders, the elicitation of fundamental requirements starts by using chosen techniques, approaches, and tools. This involves establishing the system's scope, exploring the stakeholder needs, mainly the users, and defining future processes aligned with business operations. The priority is understanding how the system can support these processes to achieve prominent objectives and address essential business problems.

Others further divide this process into additional steps, like [89], which identifies ten steps, as shown in Figure 2.1.

The requirements elicitation process has many techniques and approaches available to conduct the various activities involved in the entire process. The recognition of the importance of this process led to a growing investigation on developing techniques for improving the elicitation process [88]. There are many types of elicitation techniques, and they can be traditional, collaborative, prototyping, modeling, cognitive, contextual, and agile. Pacheco, García, and Reyes [61] analyzed the maturity of techniques most employed in the elicitation process. The ones considered in this dissertation are the ones that were proven to be the most effective. These techniques are:

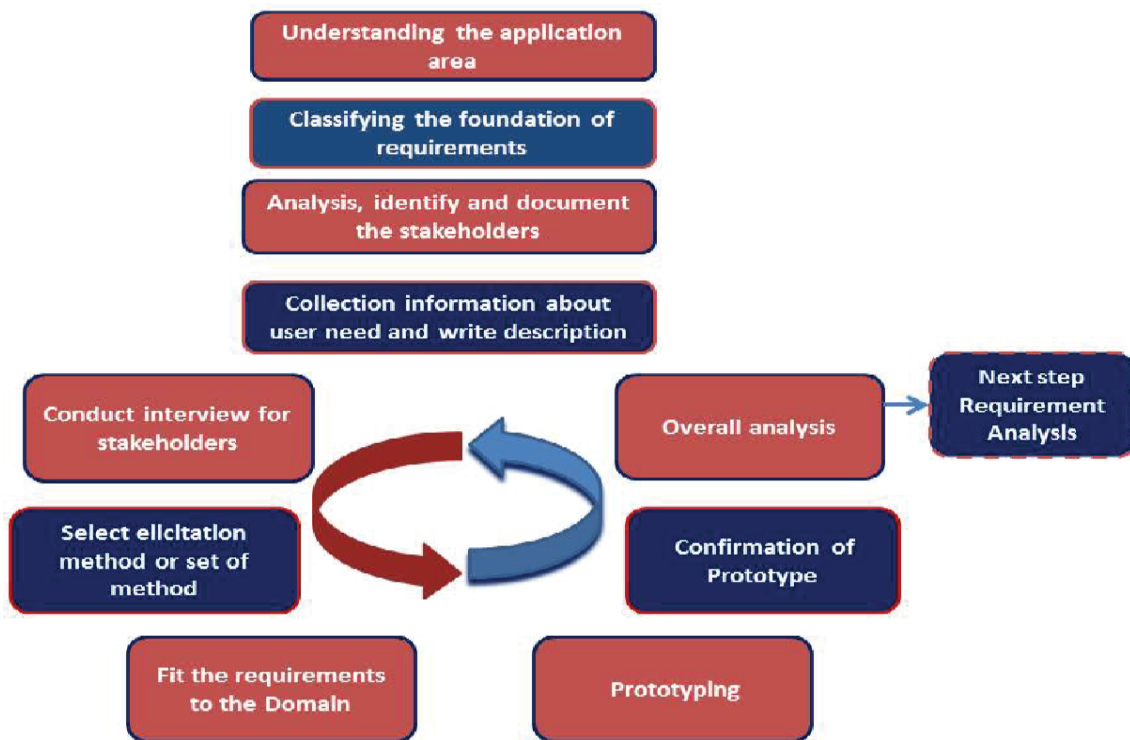


Figure 2.1: Ten Steps Requirements Elicitation Process from [89].

- **Traditional Techniques**

- *Interviews*: Interview is a method where face-to-face conversation is required to learn the opinions of users and stakeholders of the developing system. There are two types of interviews: open and closed. The open interview is where an unspecified conversation happens between the stakeholder and the requirements engineer to know the expectations for the system. The closed interview is where a group of questions are pre-selected by the requirements engineer [94].
- *Questionnaires*: A questionnaire is a technique to gather information from numerous people for less time and cost. As long as the questions are well formed, it's possible to collect the needed requirements from the stakeholders without ambiguous responses. The information obtained must have quality, i.e., not redundant or irrelevant. These questions can be open or closed, or both. This technique doesn't permit clarification or correction of misinterpretations if the participants have doubts [88].

- **Collaborative Techniques**

- *Workshops*: A workshop is a technique targeted towards the gathering of multiple perspectives. A workshop is where stakeholders assemble to debate the requirements of a developing system. When the elicitation involves a large

number of requirements, a workshop is usually conducted with a limited number of participants, involving only stakeholders directly impacted by the system [59].

- *Focus groups*: Focus group is a technique that groups four to nine people with diverse backgrounds to discuss freely possible concerns that they may have about attributes of a future system [94].
- *Brainstorming*: Brainstorming is a technique to create new ideas by grouping stakeholders from different areas. It's an informal discussion where every opinion and spontaneity is accepted, forming new requirements [59].

- **Prototyping Technique**

- *Prototyping*: Prototyping is a technique built on the communication between the experts and end-users to get feedback from the latter. The creation of prototypes occurs using initial requirements or examples of matching systems. Prototyping is more used in human-computer interfaces or when stakeholders don't know the solutions available [88].

- **Modeling Technique**

- *Scenario-based techniques*:
 - * *Scenarios*: Scenarios are narrative descriptions of processes concerning user-system interactions. Similar to use cases, scenarios don't consider the internal system structure. Their development is step by step, evolving and changing as additional information is acquired or the system develops, noting every exception in each step. Scenarios are helpful for understanding, validating requirements, and test case development [108].
 - * *Use cases*: Use cases are abstractions of scenarios detailing the functional behavior of a system. They have limitations regarding their precision, but the fact that makes them such a popular choice is their representation in diagrammatic and tabular formats that facilitate their understanding. Another point in favor is their adaptability to context-specific information. Use cases are mainly useful in situations where the analyst is not a specialist in a domain or when a project is in a stage of uncertainty [108].
 - * *User stories*: The method user stories have a format designed for capturing changing user requirements in agile software development. User stories are written in a natural language format that follows a precise structure and has information about the functionality: the entity desiring it, its necessity, and its definition. Stakeholders can interpret in various ways this method, causing misinterpretations in the development team [30].
- *Goal-oriented*: Goal-oriented uses goals for "eliciting, elaborating, structuring, specifying, analyzing, negotiating, documenting, and modifying requirements"

[101]. Goals are statements formulated at different levels of abstraction and express what the system has to accomplish. There are two distinct types of requirements: functional and non-functional. Functional requirements are related to the future services offered, and non-functional requirements are about the quality of the service, e.g., safety. KAOS is an example of notation [101].

- *Object-oriented*: Object-oriented modeling creates visual representations of real-world objects to understand problems, develop documents, and generate well-designed programs. This possesses many benefits, like generating explicit requirements, robust designs, and high-quality systems. The models must represent with as much accuracy as possible the different views of the system. Verification occurs to guarantee that the requirements match the customers. Then, the models can be translated into source code [91].

2.2 Human Values

Not everyone gives the same weight to the same human values or categorizes them the exact way. Different viewpoints or cultures can shift how they are perceived. For example, in Saudi Arabia, a woman needs to be married to be able to give birth without that being a crime [35], while in other countries, that isn't the case. Displays of affection that are a significant aspect of human development are viewed as devious in some parts of the world [93], while in others, a kiss in broad daylight is nothing to be ashamed of.

Knowing all this, how are human values categorized? In 1973, Rokeach defined values as "an enduring belief that a specific mode of conduct or end-state of existence is personally or socially preferable to an opposite or converse mode of conduct or end-state of existence" [81]. Human values, like privacy, integrity, and diversity, are given a higher level of importance in developing software than ever before, and still, this is not enough. Many more values should be considered or paid more attention to [34]. Hussain et al. [34], in 2022, defined human values in software engineering as "a set of life goals reflected in/through the software that are universally expected by civilized societies (e.g., accessibility, privacy, security, social justice) and/or personally desired by the developers and/or users and other stakeholders (e.g., happiness, achievement, social recognition)".

In this day and age, software systems are everywhere, and our world relies every single day on technologies, for instance, mobile apps, banking systems, video games, and many others. The world is a giant place where a vastly diverse number of people reside. Everything designed for human use is to bring user satisfaction, enhance comfort, and target specific needs or desires. The main goal is to create software systems that contribute to a positive and fulfilling experience for humans. Understanding that designs for human use should improve satisfaction, comfort, and overall well-being, is it that surprising to expect software developers to incorporate human values into systems? The

answer is no. The fact is that software systems are required to recognize, respect, and conform to the unique human values of their diverse end-users, including individuals with distinct characteristics such as aging or visual impairments. Therefore, to avoid harmful side effects and problems for software organizations, a non-neglectable process is operationalizing human values in software [87]. Shahin and others define this concept as "the process of identifying human values and translating them to accessible and concrete concepts so that they can be implemented, validated, verified, and measured in software" [87].

Shalom Schwartz made notable contributions to integrating human values in software, disclosing valuable insights to understand values and their applicability in the design process. Schwartz’s Theory of Universal Values is the leading conceptualization of human values that defines ten distinct basic values characterized by diverse goals and motivations [86]. These values can be further divided into sub-values, as illustrated in Figure 2.2. Table 2.1 presents the motivational goals of each value.

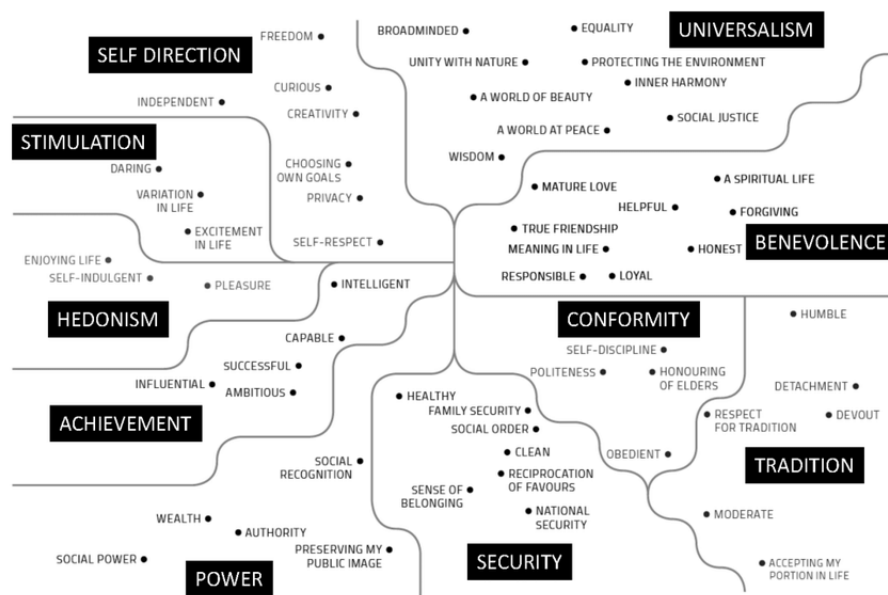


Figure 2.2: Schwartz’s theory of basic values from [34].

2.2.1 Relations between values

The Values Theory organizes values by examining their motivational similarities and differences, establishing a structure based on their relations. Any action to achieve a value leads to psychological, practical, and social consequences. These can either align or conflict with the pursuit of other values. The structure has two dimensions: self-enhancement vs. self-transcendence and openness to change vs. conservation. Figure 2.3 illustrates the relations between the values. Values that have matching motivations stand near each other. Values that are different stand as far away as possible in the circle [85].

Table 2.1: Schwartz Values, adapted from [86].

Value	Description
<i>Self-Direction</i>	Independent thought and action—choosing, creating, exploring.
<i>Stimulation</i>	Excitement, novelty and challenge in life.
<i>Hedonism</i>	Pleasure or sensuous gratification for oneself.
<i>Achievement</i>	Personal success through demonstrating competence according to social standards.
<i>Power</i>	Attainment of social status and prestige, control or dominance over people and resources.
<i>Security</i>	Safety, harmony, and stability of society, of relationships, and of self.
<i>Conformity</i>	Restraint of actions, inclinations, and impulses likely to upset or harm others and violate social expectations or norms
<i>Tradition</i>	Respect, commitment, and acceptance of the customs and ideas that one's culture or religion imposes.
<i>Benevolence</i>	Preserving and enhancing the welfare of those with whom one is in frequent personal contact.
<i>Universalism</i>	Understanding, appreciation, tolerance, and protection for the welfare of all people and for nature.

The dimension of self-enhancement vs. self-transcendence is about focusing on oneself or not. The power and achievement values, belonging to the former dimension, have the focal point of self-interest, while the universalism and benevolence values, associated with the latter dimension, concentrate on caring about the interests of others [85].

The dimension of openness to change vs. conservation has a connection to pursuing stability or embracing change. The stimulation, self-direction, and hedonism values, connected with the former dimension, the center point is independence in terms of actions, feelings, and ideas, while the security, conformity, and tradition values, related to the latter dimension, concentrate on limitations, order, and reluctance to embrace change. The hedonism value is the only one that belongs in openness to change and self-enhancement at the same time [85].

2.2.2 Factors that influence values

As defined in [85], many factors affect the values of a person. However, it's imperative to remember that values do not depend only on life circumstances; choices made on account of values also impact the life circumstances of individuals. Actions have consequences.

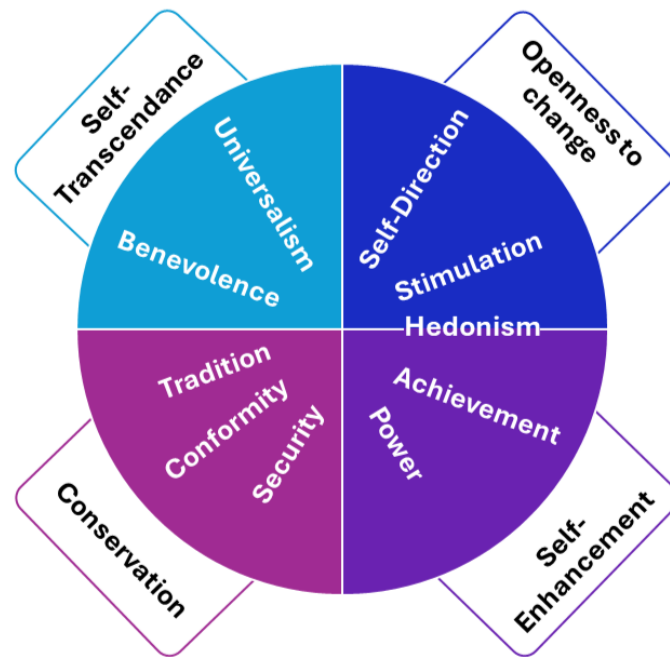


Figure 2.3: Theoretical model of relations among ten motivational types of values, adapted from [86].

The diverse characteristics in the background of somebody translate to discrepancies in life circumstances, impacting value priorities. Three prominent causes that change the values of someone are age, gender, and education. These are explained below [85].

Age. One major factor that influences values is age. Values established in adolescence tend to remain somewhat stable in later stages of life. There are three reasons values change in adulthood: historical events affecting certain age groups (such as wars or depressions), physical aging (like loss of strength or memory), and life stages (such as parenting or widowhood).

Regarding experiences of historical events, young people who lived through economic difficulties and physical insecurity grow up with materialist values more accentuated. A secure and wealthy country will cause teenagers to place less emphasis on materialistic values. This happens because, with low threats towards their way of living, they are free to explore and indulge themselves, choosing their path in life. These people will prioritize values such as hedonism, stimulation, self-direction, and universalism, while values like security, tradition, and conformity will fall behind.

With age comes unfavorable developments to someone's strength, energy, cognitive speed, memory, and sharpness of the senses. Particular values become more important, for example, security, conformity, and tradition. A safe, stable environment, as well as familiar practices, bring more comfort. Other values lose significance, like stimulation, hedonism, achievement, and power. Changes and new things are less preferred, and the reduction of a person's capacity affects their ability to perform demanding tasks.

Life stages greatly influence values, with early adulthood highlighting achievement and stimulation because of demands in work and family establishment. In middle adulthood, loyalties to established relationships divert focus towards security, conformity, and tradition values, as opportunities for change are limited. The pre-retirement and retirement stages support these tendencies, reducing opportunities for specific values while emphasizing security and tradition values.

Gender. Gender is a factor that can't be ignored. From an ethical perspective, gender differences are explained by numerous theories. Gilligan suggests that women prioritize an ethic of care and responsibility, and men focus on an ethic of rights based on justice and fairness. Evolutionary psychologists propose women gained an advantage through in-group care, while men benefited from achieving status and power. Social role theorists attribute differences to culturally defined roles, describing women as relational and expressive and men as autonomous and assertive. These contrasts lead to hypotheses that men prioritize power, achievement, hedonism, stimulation, and self-direction values, while women emphasize benevolence, universalism, conformity, and security values.

Education. Educational experiences encourage intellectual openness, flexibility, and a broad outlook essential for self-direction and stimulation values while opposing conformity, tradition, and security values. Years of formal education positively relate with self-direction, stimulation, and achievement values, displaying a consistent progression, except for universalism values, which increase in the later years of secondary school and considerably among university participants. The connections suggest that education contributes to shaping an individual's value orientations.

2.2.3 Universalism value

As mentioned earlier, numerous values hold a strong foundation in software engineering, for example, security and privacy. These values have many regulations in place to protect people. Trust is incorporated when referring to freelancing and e-commerce software using reviewer reputation systems. Hedonism (pleasure, creativity, and social recognition) is recognized through games and social media platforms. Other values are already being integrated, like gender diversity and cultural traditions. The problem is with the overlooked values. Some are social responsibility, compassion, integrity, and social justice [34]. The lack of attention regarding universalism values in software engineering is troubling since it promotes the welfare and rights of all individuals regardless of their differences. Therefore, the universalism value is selected to address its issues.

What is the Universalism value? Universalism has the following sub-values: broad-minded, social justice, equality, a world at peace, a world of beauty, unity with nature, wisdom, protecting the environment, and inner harmony. The motivational goal of universalism is understanding, appreciation, tolerance, and protection for the welfare of all

people and nature. This goal is a broader view of the benevolence value. This motivation originates from comprehending the survival needs of individuals and groups when faced with diversity and realizing the lack of natural resources. This awareness occurs from recognizing possible life-threatening conflicts that come from the failure to treat others fairly, and neglecting environmental protection may exhaust vital resources for life. In small, isolated, homogeneous cultures, these values may not be present, but these are likely to be acknowledged to some extent in nearly all others [86].

Research shows that in collectivist cultures the value of benevolence is more prominent than universalism. They are more concerned with the welfare of their ingroup than outsiders. On the other hand, individualist cultures valorize benevolence and universalism values when responding to the needs of both ingroups and outgroups, not drawing a firm line between them [86].

Universalism values integration into software engineering practices. In summary, considering the sub-values and motivational goal of universalism, this value advocates for software systems that benefit everyone, no matter their background or abilities. Technologies that do not consider these factors will lead to the exclusion of specific individuals or groups and ethical problems. These negative implications may involve privacy concerns, discrimination, and lack of fairness. In general, universalism values in software engineering should contribute to systems that support inclusion and collective well-being and advancement.

Some examples of requirements that software could have to integrate universalism values into software are:

- **Language:** Software that permits users to choose the language that they want. While this may seem obvious, in this world, languages are numerous, and more often than not, a system does not disponibilize every single language. For example, in a system with an interface only available in English, someone with difficulties understanding this language would find it extremely hard to use the system. So that everyone can operate a system, it should provide an interface with a variety of languages.
- **Color:** Picking the colors that the system presents is a function that could be useful for individuals with color blindness, visual impairments, and cognitive and neurological conditions. For color blindness, being capable of customizing the contrast for colors that they can distinguish better facilitates the use of the software (like making it easier to read), avoiding confusion. People with cognitive disabilities may find it easier to process what they are seeing, for example, people with ADHD could prefer to change colors to minimize distractions, offering a more focused environment [90]. For neurological conditions, people with migraines or epilepsy who are sensitive to some colors, giving them the option to choose colors can reduce the discomfort.

- **Disposition:** Software that allows users to decide the disposition of its elements is helpful, especially for people with cognitive and neurological disabilities. This customization feature enables individuals to transform the software interface to their specific cognitive and sensory preferences, improving usability and reducing the risk of potential cognitive overload, guaranteeing a more inclusive user experience, adapting the software to their needs, and ensuring a more comfortable environment [1].
- **Voice:** The ability to control and get information from software through voice commands aids users with mobility issues who may find it difficult to interact with small buttons. The feature enables users to receive information audibly and perform actions without requiring them to touch the screen, keyboard, or other devices. This method is an alternative and more accessible than screen contact. People who possess visual impairments would also benefit from this function [7].

2.3 Systematic mapping studies

Systematic mapping study is a method that is useful in software engineering, mainly in research areas with low-quality studies. This method organizes and structures published research reports, offering a visual summary, known as the map, that provides a more generalized perspective with less effort needed. One can identify two purposes in systematic mapping studies. The key objective is to offer a complete overview of a research area, exposing the quantity and types of research and results. Typically, throughout the years, an analysis of the frequencies of publications is done to identify tendencies. Recognizing the platforms where research in the field was published might be another minor objective [66].

An alternative method is the systematic literature review used as "a means of identifying, evaluating and interpreting all available research relevant to a particular research question, or topic area, or phenomenon of interest" [11]. It's important not to confuse the two methods. Systematic mapping studies and systematic literature reviews are similar regarding searching and study selection, yet they diverge in objectives. Systematic reviews synthesize evidence, assessing its strength, while systematic maps focus on organizing a research area [65].

The systematic mapping study process consists of five steps. The first one is the definition of research questions, i.e., the research scope, followed by conducting the search for all papers. Next is the screening of the relevant papers, then the data extraction, and finally mapping [66].

Definition of research questions

The research questions should mirror the objectives of systematic mapping studies, and they can be general and separated into multiple sub-questions. The definition of the

research questions is critical in systematic mapping studies because they are the starting point of the search process that identifies the primary studies where it's carried out the extraction and analysis of their data to get the answers to the research questions. With the answers, we get knowledge of the existing approaches to the subject at hand and any information gaps [38].

Research questions may be structured with the PICOC method that organizes the questions concerning population, intervention, comparison, outcome, and context. Determining these five criteria facilitates the systematic mapping studies and formulation of the research questions [38]. Clarifying briefly these criteria in the following list:

- **Population:** Group of interest for the investigation.
- **Intervention:** Software methodology, tool, technology, or procedure used to handle a problem.
- **Comparison:** Software methodology, tool, technology, or procedure to which the intervention is compared to.
- **Outcome:** Important conclusions of the investigation.
- **Context:** The investigation's context contains an ample perspective on the population and where the comparison happens.

After defining the research questions comes the process of conducting the search for all papers.

Conducting the search

The goal of this step is to collect a wide range of studies. Primary studies are found with search strings on scientific digital databases and manually browsing conference proceedings or journal publications. Creating search strings that are general and devised to generate many results from various databases occurs by taking keywords from each criterion of the PICOC structure. The resulting papers from the search strings and manual search form a list of every significant paper to be analyzed more in-depth in the next stage [66].

Screening of relevant papers

In this stage, the papers obtained previously through manual search and the search strings that are not relevant, i.e., that do not provide an answer for the research questions, are excluded. To this end, a set of inclusion and exclusion criteria are determined [66].

Before this stage, deciding the criteria is critical to remain impartial, but modifications are possible in the search process. Criteria based on the research questions guarantee reliability in classifying the studies without errors. In short, any paper is included as long as they have information that may answer the research questions and those that don't are excluded [38].

To properly analyze a study, the first step is reading the study's title and abstract to verify if it relates to the research questions. Any study immediately excluded saves time from an in-depth reading. There are cases where the abstract lacks quality and doesn't provide enough information to make a decision. In such cases, further reading of any part of the full text is needed to clarify any doubt. While this may prolong the time spent in this step, it raises the validity of the results. After selecting the studies, the next step is extracting the data to answer the research questions [66].

Data extraction and Mapping

For this phase, the relevant studies are organized as per the classification scheme, and then the data extraction occurs. A data extraction form contains the information. This form saves not only basic information such as the paper's title, author, journal, and publication date but also information about the research questions and the category of the paper in the scheme. With this, the frequencies of publications for each category are determined to find out if there are any gaps and possibilities for future research. The mapping of studies can be presented graphically, for example, using bubble plots. The results lack a thorough analysis of the main findings but provide a general appraisal that has sufficient information to assess the scope of a specific research area [66].

2.4 Summary

This chapter offers a rundown of the fundamental areas of study relevant to the development of the dissertation.

Starting with an integral concept in software development, we explained the requirement engineering process and the importance of requirements. This process has five main phases: elicitation, analysis and negotiation, documentation, validation, and management. Afterward, we explored the requirements elicitation process, investigating its five main standard activities on top of various available techniques. These standard activities include understanding the application domain, identifying sources of requirements, analyzing stakeholders, selecting the techniques, approaches, and tools to use, and eliciting requirements from stakeholders and other sources. The techniques studied include interviews, questionnaires, workshops, focus groups, brainstorming, prototyping, scenarios, and user stories.

Subsequently, we approached human values by discussing their meaning and implications in contemporary society. We explained the dimensions that comprise the values, motivations, and relations. Regarding the dimensions, there exist two: self-enhancement vs. self-transcendence and openness to change vs. conservation. Next, we elucidate the factors that impact values, such as age, gender, and education. To finish, we explain better the universalism value to whom this dissertation relates.

Lastly, we described the method used in the next chapter, named systematic mapping study. This method's objective and process are clarified to understand how to apply it.

The process is composed of five steps that are: definition of research questions, conducting the search for all papers, screening of the relevant papers, data extraction, and mapping.

RELATED WORK

This dissertation highlights the importance of universalism values in software engineering. To ensure that as much information as possible is found, the method used is the systematic mapping study method to conduct the research. The goal of this research work is to obtain a thorough analysis of state-of-the-art universalism values, like social justice, and how their integration takes place in software engineering. Firstly, the search process is exposed, this process is where the planning, conducting, analysis, and mapping phases are described. Secondly, a discussion is presented to resume the findings and conclusions reached. Finally, the possible threats to validity are explained.

3.1 Search Process

The systematic mapping study is helpful in situating universalism values in software engineering. The purpose is to know if they are considered in software, and if so, what way they are addressed. In the end, the results gained will present the methods and techniques that exist to integrate human values, specifically universalism values, in software development. For this reason, the mapping study method was applied in three stages: Planning, Conducting (which included the screening of relevant papers), and Analysis and Mapping.

3.1.1 Planning Phase

The planning stage is the first stage of the systematic mapping study. This stage is divided into various sections below, each described in detail.

Research Question

The method implemented is the PICOC to define the five elements to formulate the research question, as described in Table 3.1. The elements are population, intervention, comparison, outcome, and context. The method is explained in Chapter 2.3.

Following the PICO analysis and taking into consideration the keywords highlighted in it, the research question that arose is as follows:

Table 3.1: PICOC Analysis.

<i>Population</i>	Articles/papers that address the Universalism value and its approaches in Requirements Engineering .
<i>Intervention</i>	Searching online databases using appropriate strings with relevant keywords related to Universalism in Requirements Engineering.
<i>Comparison</i>	There will be no comparison.
<i>Outcome</i>	Analysis of the state of the art on the investigated subjects.
<i>Context</i>	Preparation for Master Thesis in Computer Science and Informatics Engineering.

How are Universalism and its sub-values addressed in the elicitation and specification activities in Requirements Engineering?

Since this research question is too general, the main question is unraveled into three sub-questions that will help guide a more focused and detailed search on the subject under examination. Table 3.2 shows the sub-questions and respective motivations behind each one.

Table 3.2: Research Sub-questions.

Research Sub-question	Motivation
<i>RQ1: What are the existing approaches for elicitation and specification for Universalism?</i>	To learn the specific approaches or methods used in the process of gathering information (elicitation) and the process of documenting or defining requirements (specification).
<i>RQ2: What are the existing tools for elicitation and specification for Universalism?</i>	To learn current tools for addressing Universalism values in requirements engineering.
<i>RQ3: What are the topic areas where Universalism values were addressed?</i>	To learn the topic areas where universalism values were addressed.

Search Process

The search process objective is to identify primary studies. For this purpose, there are two search methods: Automatic and Manual. A manual search was conducted by selecting papers that appeared relevant throughout the process and while researching the human values topic, backward snowballing was employed. Regarding the automatic method, this was executed by creating search strings with keywords retrieved from concepts that

originated from the sub-questions. Table 3.3 indicates the keywords obtained from the respective concepts.

Table 3.3: Search Keywords.

Concepts	Keywords
<i>Universalism</i>	universalism AND ("social justice" OR equality OR wisdom OR broadminded OR "a world of beauty" OR "a world at peace" OR "unity with nature" OR "inner harmony" OR "protecting the environment")
<i>Approach</i>	approach OR method OR process OR technique OR model OR tool
<i>Software</i>	"requirements engineering"
<i>Elicitation</i>	elicit OR specification

The Google Scholar database was chosen to run the search strings as queries. Google Scholar is a digital library that simplifies the search for scholarly literature from numerous sources, for example, academic publishers, professional societies, online repositories, universities, and other websites [27]. Some examples of sources are ACM, IEEE, and Springer. The search strings are formed with the keywords from each research sub-question. These keywords are connected with the logic operators AND and OR. Words that should appear together, like social justice, are enclosed in quotation marks. With the search strings, the aim is to capture all the resulting works related to the keywords. In total, for each sub-value from universalism, it's possible to define nine search strings, however only five were used. Below are the five queries, and then a justification for the choices is given.

1st Search String: universalism AND ("social justice" OR equality) AND (approach OR method OR process OR technique OR model OR tool) AND "requirements engineering" AND (elicit OR specification)

2st Search String: universalism AND broadminded AND (approach OR method OR process OR technique OR model OR tool) AND "requirements engineering" AND (elicit OR specification)

3st Search String: universalism AND "a world of beauty" AND (approach OR method OR process OR technique OR model OR tool) AND "requirements engineering" AND (elicit OR specification)

4st Search String: universalism AND "a world at peace" AND (approach OR method OR process OR technique OR model OR tool) AND "requirements engineering" AND

(elicit OR specification)

5st Search String: universalism AND “inner harmony” AND (approach OR method OR process OR technique OR model OR tool) AND “requirements engineering” AND (elicit OR specification)

Why these queries? First, to narrow the scope, the *universalism* keyword has to be obligatory, otherwise, too many results are obtained, and it wouldn't be fallible to analyze all of them. Then, for the first search string, the sub-values *social justice* and *equality* are investigated together since in all the sub-values, they are the subjects more related to one another and with the most results. In the following search strings, the sub-values are either too abstract or have little to no information found on the subject. So, it was decided to separate the sub-values into different search strings to get an overview of the studies acquired for each subject. The sub-value *wisdom* isn't explored due to the abstract nature of the word. Many of the results have no relation to the topic in question, and the few that appear to be are already investigated in the other queries. Finally, the sub-values of *unity with nature* and *protecting the environment* aren't researched, because while their subject is related to sustainability, an important aspect to consider in software and part of the universalism value, it is a complex area and a world on its own, so the conclusion reached was that the focus of this dissertation would remain on the other sub-values.

Selection Process

After defining the search strings, it's time to analyze and evaluate all the studies found. To accomplish this, it's important to establish a set of inclusion and exclusion criteria that will contribute to selecting the appropriate papers. This process is necessary since not every study captured from the search process is relevant to answering the main research question and sub-questions. The defined criteria will help increase the chances of finding studies essential for extracting the desired data.

The process concluded after two iterations. Table 3.4 presents the criteria specified for the first iteration of the study selection, and Table 3.5 represents the criteria for the second iteration. During the first iteration, the title and abstract of every study were read and reviewed. Those who fulfilled all the inclusion criteria, lacking any exclusion criteria, were moved to the next phase. The second iteration is where a complete reading of the pre-selected studies from the first iteration occurred. The ones that satisfied the inclusion criteria, without any exclusion criteria chosen, remained and became the final primary studies.

Quality assessment

When finished with the selection of the papers, it's necessary to assess their quality. For this end, the CORE Rank measure was utilized. The CORE Conference Ranking assesses major conferences in the computing disciplines. Depending on various factors,

Table 3.4: Inclusion and Exclusion Criteria for the first iteration.

ID	Inclusion Criteria
IC1.1	The article is available in English.
IC1.2	The article was published in conferences, workshops and journals.
IC1.3	The article is available in full text.
IC1.4	The title and/or abstract of the paper is related to at least one of the research questions (RQ1, RQ2, RQ3).
ID	Exclusion Criteria
EC1.1	The article is a duplicate.
EC1.2	The article isn't in English.
EC1.3	Articles in an abstract, tutorial, poster, or presentation structure.
EC1.4	Article is not available in full-text.
EC1.5	The title and/or abstract of the article do not relate to any of the research questions (RQ1, RQ2, RQ3).

Table 3.5: Inclusion and Exclusion Criteria for the second iteration.

ID	Inclusion Criteria
IC2.1	The article's content has information about one or more research questions (RQ1, RQ2, RQ3).
ID	Exclusion Criteria
EC2.1	The article's content does not answer any of the research questions (RQ1, RQ2, RQ3).
EC2.2	The article's content discusses challenges related to the main research question but does not provide a solution, or it proposes measures but does not explain how they are used.

different categories are assigned: A* is a flagship conference, A is an excellent conference, B is a good to very good conference, and C is every other conference that meets the basic standards [13]. Another factor to consider is the number of citations. The number of citations generally means that if a study is highly referenced, then it's trustworthy and more reliable than a study with fewer citations. The year the study was published is also a factor to consider since a newer study may have fewer citations because it's not yet well known. None of the studies were excluded based on quality assessment. However, in the future, if there is any conflict between studies, like a concept that is contradicting between studies, then the quality will be taken into account when deciding which study to follow.

Data collection and extraction

To address the research questions, in the Excel worksheet where every study was evaluated during the second iteration, columns were added to document the information needed to facilitate the analysis. Every single paper had its content extracted using the same criteria. The worksheet had basic details of the paper, such as title, author(s), and a section with relevant information about the research questions. Table 3.6 presents the information we intended to extract.

Table 3.6: Data extracted into the worksheet.

Paper Basic Information
ID
Title
Author
Year
Conference/Journal/Workshop
Digital Library
Quality
Number of Citations
CORE Rank
Data extracted for the research questions
Goal of the study
Important requirements for Universalism in RE
Elicitation and Specification techniques for universalism (models, methods, tools)
Software application domains
Weaknesses and Strengths
Validation and/or Evaluation process of study

3.1.2 Conducting Phase

Table 3.7 provides the results of the data extraction process. Every query is present, with the number of papers obtained before any selection and the discarded number for each exclusion criteria. In the end, it's revealed the total number of papers accepted.

Synthesizing this process, query number one has the most results, with 121 papers, while the others don't even reach 10. After the first iteration, only query one has remaining papers, 24. The rest of the queries mostly had papers duplicated found in the first query, and the 2 remaining papers from the second string were not relevant. In the second iteration, after analyzing in depth the papers, only 4 remained. So, from the automatic search, the number of final studies was 4. A manual search was conducted to complement

Table 3.7: Results of the automatic search.

Query	1st	2nd	3rd	4th	5th
1^oMatches	121	8	5	6	6
EC1.1	-9	-6	-5	-6	-6
EC1.2	-1	0	0	0	0
EC1.3	-40	-2	0	0	0
EC1.4	-27	0	0	0	0
EC1.5	-20	0	0	0	0
2^oMatches	24	0	0	0	0
EC2.1	-17	0	0	0	0
EC2.2	-3	0	0	0	0
Total	4	0	0	0	0

this process, and from this 6 more papers were found using snowballing and keyword manual search. Concluding, the number of studies from the automatic and manual search was 10. Table 3.8 presents the final studies selected.

Table 3.8: Titles of the 10 papers selected in the automated and manual search.

Reference	Study Title
[2]	Building social sustainability into software: Case of equality
[48]	Accountable Design for Individual, Societal, and Regulated Values in the UAV Domain
[40]	The implementation of enterprise resource planning packages in different organisational and national cultures
[104]	Human Values as the Basis for Sustainable Software Development
[57]	GIRE: Gender-Inclusive Requirements Engineering
[42]	Evaluating social sustainability in historic urban environments
[52]	A Conceptual Model For Web Accessibility Requirements In Agile Development
[63]	A Generic Model for Sustainability with Process- and Product-specific Instances
[62]	Towards a Catalog of Privacy Related Concepts
[19]	Specifying Reusable Security Requirements

3.1.3 Analysis and Mapping Phases

This phase will analyze the studies found in the conducting process. This section has three subsections, one for each research question, to provide an answer elucidating the findings of the studies.

RQ1: What are the existing approaches for elicitation and specification for Universalism?

The approaches for requirements elicitation are many, however, discovering any for universalism values was difficult since for all the articles found from the previous processes, not one of them had an approach specifically for universalism values. Some only address specific sub-values from universalism, such as equality, sustainability, gender, and accessibility. So, in this research, there were studies that provided, for some sub-values, models and frameworks ([2], [57], [52], [63], [62], [19]), an evaluation ([42]), a list of values ([104]), and a meta-schema ([40]). Study [48], which covers human values, didn't present any approach to elicit requirements and used a survey.

The models and frameworks identified each refer to specific sub-values of universalism. The study in [2] presents an Equality Value Pattern to identify important software requirements related to equality. The study in [57] offers a conceptual model for gender-inclusive requirements, to simplify gender analysis in the elicitation process. The study in [52] proposes a conceptual model for web accessibility to guide the integration of accessibility considerations into agile software development. The study in [63] suggests a generic sustainability model to support and assess environmental sustainability in software engineering. The study in [62] presents the privacy conceptual model that contributes to creating the basis to standardize concepts related to privacy. The study in [19] presents both a model with concepts that are influenced by security requirements and a taxonomy of security quality factors and subfactors.

The study in [42] offers a social sustainability evaluation criteria for historic urban environments. The study in [104] presents a list of values to foster sustainable software development by allowing engineers to define dimensions of sustainability using values. Finally, the study in [40] provides a knowledge meta-schema that outlines the organizational and cultural elements influencing ERP implementation.

RQ2: What are the existing tools for elicitation and specification for Universalism?

The number of studies selected after the two iterations was 4, and even these did not provide any tool for universalism values. With the manual search, it was possible to find one study ([57]) that developed a tool to support the method developed. The study in [57] created the GIRE tool that supports the integration of the model of gender-inclusive requirements in requirements elicitation. The lack of tools indicates that, at the moment, there is very little in requirements elicitation about universalism.

RQ3: What are the topic areas where Universalism values were addressed?

The topic areas of the studies were diverse, but mainly were in the requirements engineering domain ([2], [48], [57], [63], [62]), which is logical given that the research strings had the keyword "requirements engineering". The Human-Computer Interaction (HCI) domain was incorporated in study [2], while the process engineering domain was also included in study [63]. The study in [57] was specifically for gender-inclusive software.

The rest of the studies were in various other topic areas: Enterprise Resource Planning (ERP) systems ([40]); Sustainable software development ([104]); Social sustainability domain ([42]); Agile software development ([52]); Security engineering ([19]). Two studies that demonstrated and explained with case studies had the following topic areas: Web-based applications ([2]) and Emergency response systems ([48]).

3.2 Discussion

With this systematic mapping, it's clear that besides the sub-values *equality* and *social justice*, not counting *unity with nature* and *protecting the environment* that, for reasons already explained, will not be considered, the rest of the values are overlooked in the integration in software. Connecting the ideals of the *broadminded* and *wisdom* sub-values to *social justice* and *equality* becomes possible when we consider their innate compatibility. Broadminded is all about being open to new ideas, changes, experiences, and perspectives. In *social justice* and *equality*, this would translate into accepting different backgrounds and viewpoints. This is the concept of inclusion that promotes an environment where people are respected and treated fairly. Wisdom can translate into understanding human nature and the skill to make correct choices. In *social justice* and *equality*, wisdom can direct individuals and society to acknowledge the existence of injustices, social problems, and others. So, even if there weren't any studies directed toward these other sub-values of universalism, they all connect in some way.

In general, we can conclude that some sub-values (like equality) are more addressed than others (like inner harmony), and there doesn't seem to exist a method, model, or tool, that effectively integrates the universalism value. What exists in software engineering tends to lean towards human values as a whole or just one sub-value in particular, as the systematic mapping studies demonstrated. In closing, this dissertation will focus more on the sub-values of *equality* and *social justice*.

3.3 Threats to validity

In an attempt to prevent threats to the validity of this study, certain actions were taken to mitigate them and increase the correctness of the results. Nonetheless, a study always has limitations related to the methodology chosen for the research. One problem is

the accuracy of the automated search. In systematic mapping studies, some studies significant to the context escape the process. To avoid this issue, the search strings created were optimized with keywords relevant to the research questions, but unfortunately, the outcome may be incomplete or inaccurate because there is no way to guarantee that all studies use the keywords. Another possibility is the query itself, which might be restrictive. With these problems apparent, a manual search was conducted to lessen this threat and increase the number of papers important to the dissertation. An additional concern is to prevent the study selection from being biased, as the study selection was executed by one person only, and the number of studies was not small, some studies could be overlooked. For this reason, a list of inclusion and exclusion criteria was defined to ease the study selection. Furthermore, the digital library was carefully chosen to provide an extensive collection of studies from diverse sources. Google Scholar was preferred for this purpose. Another matter worth mentioning is that any non-English or unavailable study wasn't considered.

3.4 Summary

This chapter implemented the method of systematic mapping studies explained in Section 2.3. First, we started with the search process, where the research questions and queries were defined so that, in the next step, the studies could be selected after a series of criteria. Then, the data was extracted and analyzed to answer the research questions. Afterward, a brief discussion was presented with the overall results summarized and the conclusions obtained. We finished this section with the threads to validity that are always a risk of existing in a study.

CONCEPTUAL MODELING OF UNIVERSALISM REQUIREMENTS

Universalism values, such as equality, social justice, broadmindedness, and wisdom, encourage a fair and inclusive society. Human life revolves more and more around software, increasing the necessity to explore the integration of universalism values in software development.

In Chapter 3, the systematic mapping study conducted on universalism values in requirements engineering confirmed that this area of research is essential. The study results revealed that while equality and social justice receive attention, there is a lack of tools to address these values during software development.

In this chapter, we propose a conceptual model of universalism requirements. The findings of the systematic mapping study served to identify and define the concepts and relationships that compose the model. The objective of this work was to handle the gaps existing in the field, mainly the lack of frameworks and tools addressing universalism values, starting by developing a definition and a conceptual model to standardize the concepts to facilitate the integration of universalism values in the requirements elicitation process so that software systems reflect these values.

4.1 Building the conceptual model

The development of the conceptual model followed an incremental three-step process based on the methodologies proposed in [18, 56]: Specification, Knowledge Acquisition, and Conceptualization. This ontology-based approach was selected because it helps organize and define key concepts and their relationships in a clear and structured way. The following sections clarify each phase of the process.

4.1.1 Specification

The specification phase consists of two steps: the first relates to identifying the domain and purpose of the conceptual model, and the second defines its scope. In the first step,

we explain why the conceptual model is required, what it will be used for, and by whom. For this, we describe the motivation for developing the model, use scenarios where it applies, and its range of intended users, written in natural language. In the second step, we determine the concepts and terms that fulfill the conceptual model requirements. The next subsections present the results of both steps.

Domain and Purpose

The conceptual model domain relates to the integration of universalism values in the design and development of software. The purpose is to help develop software that respects universalism values, which encompass values of equality, social justice, broadmindedness, and wisdom. The model supported the construction of a framework that assists the understanding of these values and their importance in the software context.

The model is most useful for requirements engineers, software engineers, and stakeholders to elicit universalism requirements for software applications and systems, ensuring fairness and inclusiveness within the software.

Scope

The conceptual model scope determines the concepts that are key in representing universalism values in software. In this case, the values considered specifically are equality and social justice for reasons already explained in Chapter 3. So, based on the information of the papers found, we defined four main categories: *Equality*, *Security and Privacy*, *Culture*, and *Gender*. These are the foundation of the conceptual model and represent significant aspects of universalism values.

The model is directed towards software whose goal is to foster social justice and equality and support community welfare, although the model is usable on a wide range of applications. For example, systems that promote digital inclusivity or aid underrepresented communities. The objective is to guarantee inclusivity and equality in software.

4.1.2 Knowledge Acquisition

The knowledge acquisition phase focuses on identifying, gathering, and defining the knowledge needed to build the model according to its purpose. We started by collecting unstructured, semi-structured, and structured knowledge regarding universalism values in software development. We identified the main concepts from the resulting papers of the systematic mapping study in Chapter 3, which helped us understand how universalism values interact with software. For each concept in the model, the definitions were taken from the papers or from reliable online sources, such as dictionaries or papers that were not part of the list from the study. When needed, we refined the definitions further to fit the software systems context by adding phrases that clarify how each concept functions in a software environment.

Then, we organized the concepts and relationships in a semi-formal taxonomy form. The taxonomy comprises six categories: *Equality*, *Security and Privacy*, *Culture*, *Gender*, *Human*, and *Software System*. These are fundamental to the definition of the *Universalism Requirements*.

4.1.3 Conceptualization

The conceptualization phase concerns arranging the concepts and respective relationships into a structured and visual representation. The previous taxonomy is the basis for the conceptual model. The model shows the connections between concepts, such as *Human*, *Software System*, and *Security*, displaying the links between categories. We created the model's first draft with the data compiled. This process suffered continuous evaluations, alternating with the previous phase to review and refine the concepts' definitions and relationships.

One key goal for this phase was to ensure the definitions of the universalism-related concepts were clear to increase comprehension for people involved in the software development process, while at the same time preserving the complexity needed to comprehend the concepts. We built the model using UML, a notable modeling language in software engineering.

4.2 A conceptual model for Universalism requirements

This section describes the conceptual model for universalism created as a UML call diagram, shown in Figure 4.1.

The model comprises the six main parts below:

- **Equality:** Covers social equality, socio-cultural equity, and fairness. *Social Equality* secures equal access to services and opportunities for all users, while *Socio-cultural Equity* involves concepts that promote inclusivity regardless of individual characteristics or abilities. The *Fairness* concept highlights equitable treatment by ensuring software systems provide equal opportunities without bias. These principles support *Social Sustainability* by encouraging fair resource distribution and opportunities.
- **Security and Privacy:** Covers the concepts of security, privacy, and trust. *Security* includes elements such as *Security Mechanism* and *Asset*. An asset is anything valuable that must be protected from attacks, and for this purpose, security mechanisms are implemented to prevent and handle any intrusion. *Privacy* comprises *Privacy Mechanism* and *Personal Information*. *Personal Information* refers to the various data that could identify an individual, while *Privacy Mechanism* are tools designed to protect the confidentiality of the users' data. These also guarantee that users' best interests are considered and that their consent is requested regarding their data. All of these measures increase the *Trust* users place in software systems.

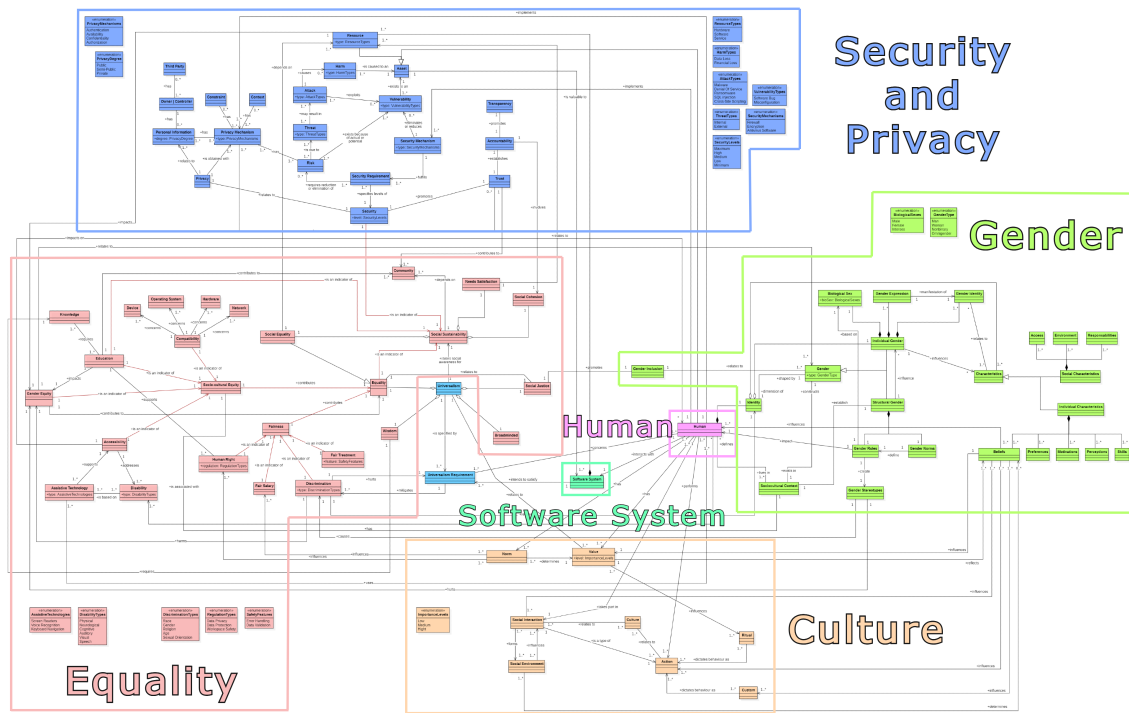


Figure 4.1: Conceptual model of universalism requirements.

- **Culture:** Covers the concepts of values, norms, and social interactions. *Social Interaction* represents the connections between users in software systems. *Value*, a belief linked to emotion, influences system design and functionalities, aligning with user needs and expectations. *Norm* is an expected behavior that users must follow to promote community guidelines and ethical standards. These concepts combined contribute towards a respectful environment that improves user experience.
- **Gender:** Covers *Gender*, which comprises *Structural Gender* and *Individual Gender*, and is influenced by the *Sociocultural Context*. *Structural Gender* refers to societal norms and opportunities assigned to individuals based on perceived sex, while *Individual Gender* reflects a person's self-identified gender. The *Sociocultural Context* highlights how social and cultural factors, such as age, gender, and race, impact the software development process and requirements.
- **Human:** Defines the concept of Humans, highlighting the individuals' diverse characteristics who interact with the software. Users have different experiences, roles, opportunities, levels of access, and identities, each shaping their expectations and interactions with software technologies. Therefore, humans expect the Software System to be inclusive and acknowledge their characteristics, providing the necessary support.
- **Software System:** Captures a software system as the product of human-based decisions influenced by the components of universalism. The system intends to

satisfy Universalism requirements by being adaptable and accessible to the needs of the different users.

The following subsections present the model parts, detailing the concepts and their relationships.

4.2.1 Equality

In this section, we explain the equality part of the conceptual model, presented in Figure 4.2. This part draws on earlier work found during the search process in Chapter 3, with most of the equality part conceptualized in [2].

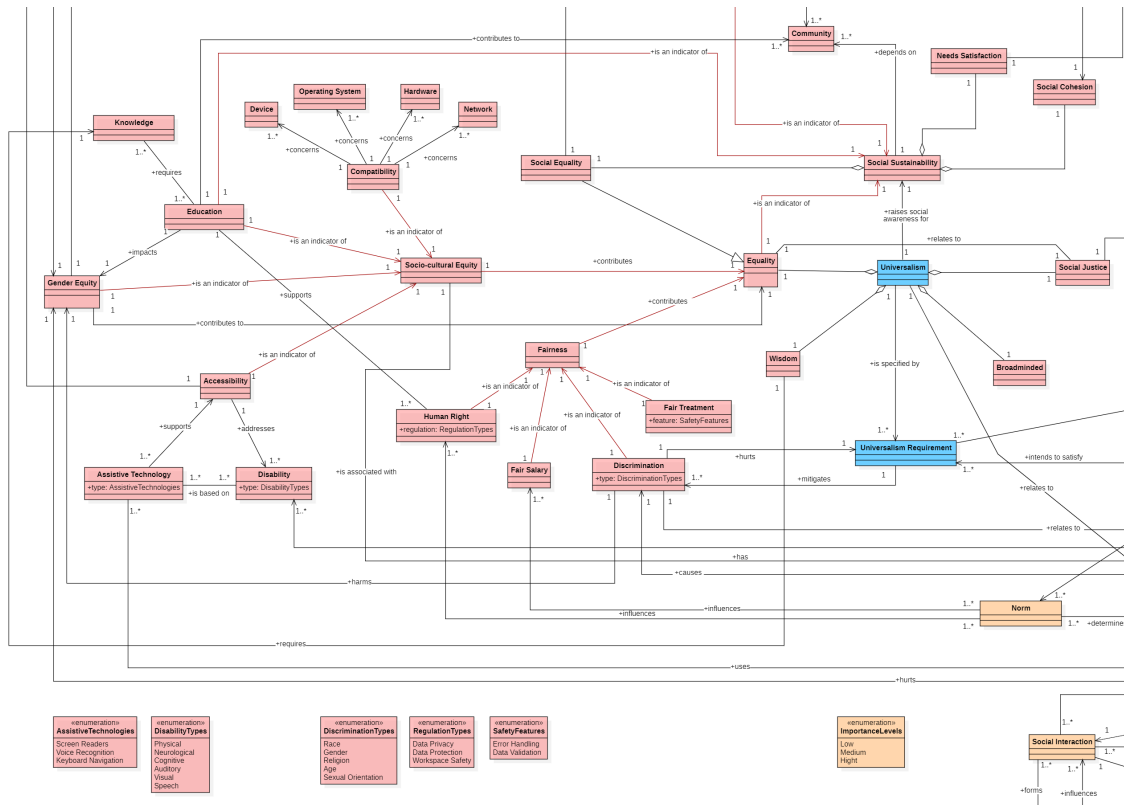


Figure 4.2: Equality part of the conceptual model.

In this day and age, software systems maintain significant control over access to information, services, and education, making it crucial for them to address equality requirements [3].

When certain groups are always deprived in favor of others, it often brings about social tensions, unrest, and even uprisings [8]. This stresses the critical need for systems to promote fairness and equal access. Regardless of the domain, the concept of *Equality* can be applied to various applications [3], ensuring that all users have the same opportunities and access without discrimination. As technology impacts everyday life, incorporating

equality into software systems prevents injustices and fosters inclusive and harmonious digital environments.

Equality, defined as “the right of different groups of people to have a similar social position and receive the same treatment” [74], is one of the sub-values of *Universalism*. In software systems, *Equality* exists in systems that ensure all users have the same opportunities and access to features and services, without discrimination based on gender, race, or other personal characteristics. This extends to fairness in the workplace through policies and practices.

Thus, in this model, *Equality* leans on the concepts of *Fairness*, *Socio-cultural Equity*, and *Social Equality* [2] to aid in developing software systems that uphold this universalism value, guaranteeing that all users are treated equally and have the same opportunities to benefit from the technology.

The *Equality* concept relates closely with *Social Justice*, another sub-value of *Universalism* described as “the idea that all people should have the same rights and opportunities and that a country’s wealth and resources should benefit everyone in that country” [76]. In other words, while *Equality* directs attention to treating individuals the same way and granting equal opportunities, *Social Justice* aims to deal with systemic inequalities and promote fairness. *Social justice* expands past the equal treatment of individuals, stressing the need to understand and dispute the deeply ingrained structures and practices that prolong inequality and disadvantage marginalized communities. Addressing these factors will help create a more inclusive and fair environment [29].

To understand the difference between these concepts, the ShareTheMeal app is used as an example to illustrate them. ShareTheMeal is a fundraising application by the United Nations World Food Programme (WFP) whose goal is to fight global hunger. It allows users to donate small amounts of money to various causes all around the world by simply using their phones. It’s designed for quick and easy use, allowing everyone to make an impact on this cause [79]. While the application ensures that every user has equal access to features like making donations, following the *Equality* concept, it also incorporates targeted donation options for specific issues (e.g., help families fleeing conflicts). Each donation option includes educational content that informs users about the problems related to these issues and the latest updates. Since the app raises awareness and fosters understanding, it contributes to *Social Justice* by addressing the heart of social inequities problems.

Together, these concepts aspire towards equal opportunities and access to resources for everyone, regardless of background or identity.

Universalism raises social awareness for *Social Sustainability*, conveying the importance of equality and community well-being.

Social Sustainability is determined as “a positive and long-term condition in communities and a process within communities for achieving and maintaining that condition” [49]. It concentrates on guaranteeing that present and future generations have equal or greater access to social resources by pursuing generational equity. In software systems, it

means supporting social communities in different domains and initiating activities that cause indirect benefits for these communities [41]. In other words, it advocates for fair and equitable access to resources and opportunities for everyone interacting with or benefiting from the system.

As an indicator of *Social Sustainability* [2], *Equality* enables the development of systems designed to meet the basic needs of diverse communities while ensuring equitable treatment for all users, regardless of their background [42]. This contributes to the physical and psychological well-being of individuals [42], reinforcing the sense of community among users.

Social Sustainability depends on a strong *Community* [63], where systems are designed to enhance community bonds and cultivate shared values. The *Community* refers to “the people living in one particular area or people who are considered as a unit because of their common interests, social group, or nationality” [70]. In software systems, community involves users connected by a digital platform, sharing common interests and goals. This includes not only the users engaging with the system but also those affected by the system’s objectives.

By addressing the dimensions of *Social Sustainability* [42], specifically *Needs Satisfaction*, *Social Cohesion*, and *Social Equality*, software systems can create environments that help individual users and the wider community.

Needs Satisfaction is important to ensure that basic human needs are met. It involves evaluating the *Resource* level and opportunities available to all community members. This satisfaction can be measured with objective satisfaction, which examines the concrete resources available, and subjective satisfaction, which focuses on individual perceptions of their needs being fulfilled [42]. In software systems, *Needs Satisfaction* assesses the resources and opportunities the systems give to the entire community of users.

Social Cohesion emphasizes cooperation and unity among community members. It is defined as the absence of social conflicts based on factors such as wealth, ethnicity, and gender, and the presence of strong social bonds such as civic engagement [36]. The characteristics of *Social Cohesion* include the strength of support networks, levels of participation, shared identity, and mutual tolerance, reflecting the empowerment and *Accountability* of the community [42]. In software systems, systems that support *Social Cohesion* translate into an online community that is inclusive and supportive, where users cooperate and interact positively.

Social Equality occurs when all community members have equal access to and benefit from the available services, facilities, and opportunities. This includes, for example, access to healthcare, education opportunities, and housing services [2]. *Social Equality* relies on *Resource* availability within the community. Without these resources, it is hard to ensure equal benefits for all community members. This concept measures how equitably resources and opportunities are distributed in a community. The main characteristics include access to services, facilities, and opportunities and the institutional stability and flexibility that ease this access [42]. In software systems, *Social Equality* ensures every

user has equal access and benefits from the system's services, leveling the playing field so every user can use the system for personal or professional growth.

The *Social Equality* concept is a specialization of the more general *Equality* concept. This means that while *Social Equality* centers on specific aspects like access to resources, it is part of the larger notion of *Equality* that encompasses the principles of fair treatment and equal rights for everyone.

The *Fairness* concept contributes to *Equality* by ensuring that all employees receive equal treatment and opportunities in an organization [2]. Organizations should give identical conditions for employees with similar job descriptions, such as equal vacation days and salaries. In software systems, fairness concerns systems designed to secure equal access and opportunities for all users. This includes algorithms that must be free from bias to give equal access to features and policies that promote equitable treatment for all users. Several indicators contribute to the concept of *Fairness* [2]: *Human Right*, *Fair Salary*, *Fair Treatment* [48], and *Discrimination*.

Human Rights are the fundamental rights and freedoms every person in the world has. These rights follow principles like dignity, fairness, respect, and equality, and they are protected by law. Employers must stick to these principles [55]. In software systems, this means allowing employees necessary rest days and safeguarding their personal data from unauthorized access.

Fair Salary is about employees receiving fair compensation depending on their contributions and responsibilities in an organization. They should be judged based on their skill and experience and the salary must reflect these aspects. In software systems, the system can't allow a person to receive more or less than someone else who does the same work just because of gender, for example.

Fair Treatment means giving all employees equal and impartial treatment, regardless of their personal characteristics. This involves fair evaluations and opportunities for promotion based on merit. Concerning software, *Fair Treatment* may involve designing systems that reduce personal liability for people in stressful situations, like forgiving mistakes in stressful moments. This might entail the creation of features that prioritize safety, for example, safety features like error handling can guarantee that users are not punished for accidental mistakes, while data validation can prevent errors before they occur, this way reducing stress [48]. These features help create a fairer, more supportive environment for individuals interacting with the system.

In this model, the concept of *Discrimination* is included as an indicator of *Fairness* to align with the need to address equal opportunities [2]. *Discrimination* is the unfair treatment of any person or group based on race, gender, religion, or other characteristics [72]. It directly relates to *Identity*, as people's characteristics or group associations can make them targets of unfair treatment. *Discrimination* harms *Gender Equity*, as it perpetuates unequal treatment based on gender, contributing to ongoing disparities. Software systems must avoid discriminatory practices, ensuring that algorithms are free from bias. The environment must promote respect and inclusion, whether in the workplace or online.

Any action that may cause harassment or exclusion of individuals in an organization or online environment will harm the *Universalism Requirement* because it damages the principle of *Universalism*. On the other hand, implementing universalism requirements will mitigate the chance of *Discrimination*.

Socio-cultural Equity contributes to *Equality* by ensuring that individuals' characteristics, such as gender, age, or race, do not affect their access to software services. It promotes inclusivity by ensuring that community members can interact with the system, regardless of the technology they use or their abilities [2]. *Socio-cultural Equity* is associated with the Sociocultural Context, which refers to the broader social and cultural environment that shapes gender and life experiences [57]. While the *Sociocultural Context* influences the system's design and development [57], *Socio-cultural Equity* directly addresses how access to technology and opportunities is accomplished across diverse user groups [2].

The indicators of *Socio-cultural Equity* are *Compatibility*, *Education*, *Gender Equity*, and *Accessibility* [2]. Each reflects the main areas where equity must be fostered to ensure inclusivity in software systems.

Compatibility refers to the ability of a software system to operate in different devices, operating systems (OS), hardware components, and networks. Additionally, it also relates to the ability of these elements to work together. It seeks to confirm that users can access the software regardless of their technology. The *Device* concept includes desktop, tablet, mobile, tablet, and other electronic devices. The *Operating System* covers Android, iOS, Mac, and Windows, while *Hardware* compatibility involves processors, memory, graphics cards, and other components. Also, *Network* compatibility extends to wired or wireless networks, LANs, WANs, or the internet, ensuring stable connectivity for all users.

Education is "the process of teaching or learning, especially in a school or college, or the knowledge that you get from this" [73]. It is a basic *Human Right* that empowers individuals and levels inequalities [96], contributing to the overall *Community* well-being [63].

Additionally, *Human Rights* support *Education* and vice versa [63], since access to education is essential for individuals to understand and advocate for their rights. When human rights are maintained, they create an environment where education is accessible to everyone. Thus, the two concepts are connected and reinforce each other to promote a better society. By safeguarding educational access, communities promote social sustainability.

Education is an indicator of *Social Sustainability* [2] and is important to improve the engagement of the individuals in the community. *Education* supplies individuals with the necessary knowledge and skills to allow them to participate in the community. *Knowledge*, defined as "understanding of or information about a subject that you get by experience or study" [75], is vital in this process. In software systems, it implies giving users information that improves their understanding of the system's purpose, functionality, or subject matter. It means delivering clear explanations, insights, or data that users can utilize to make informed decisions or actions in the system.

Moreover, *Education* impacts *Gender Equity* by providing equal opportunities for all

genders to learn and thrive. *Gender Equity* is defined as the “provision of fairness and justice in the distribution of benefits and responsibilities between women and men” [23]. In this model, we expand this concept to include all genders, advocating for equity regardless of gender, such as women, men, non-binary individuals, transgender individuals, and other gender minorities. Everyone should be allowed to learn, regardless of their circumstances, such as economic status or personal characteristics, like gender.

Gender-based expectations have a significant impact on students’ motivation, self-concept, and performance. When teachers address these biases, they create an inclusive environment. Lessons must be designed to reduce stereotypes, giving every student access to learning opportunities. Fostering gender equity in education will build a community that allows individuals to reach their full potential [95]. In software systems, *Gender Equity* indicates that all users, regardless of gender, have equal access to and benefit from the system’s functionalities. In addition, educational systems help users gain skills and knowledge by interacting with the system, creating opportunities for personal and professional growth.

Resources, including hardware, software, and system functionalities, have an impact on *Gender Equity*. Unequal access puts certain genders at a disadvantage, while fair distribution empowers everyone, promoting inclusivity.

Accessibility refers to the ability of a software system to be usable by individuals with different abilities, for example, individuals with disabilities who must rely on assistive technologies. *Disability* is “an illness, injury, or condition that makes it difficult for someone to do some things that other people do, and that is usually permanent or lasts for a long time” [71]. There are many types of disabilities, they can be Physical, Neurological, Cognitive, Auditory, Visual, and Speech related [52]. Software systems must accommodate users with different disabilities to enhance inclusivity.

Accessibility directly addresses *Disability* by improving the human experience when using systems. It accomplishes this with the support of assistive technologies designed with individuals’ disabilities in mind to allow them to perform tasks more effectively [52].

Assistive Technology “describes tools used by people with disabilities to accomplish tasks” [99]. In a web context, it relates to “hardware or software used to increase, maintain, or improve functional capabilities of individuals with disabilities” [52]. Examples of assistive technology are screen readers, keyboard navigation, and voice recognition software, indispensable for users with disabilities to interact with software systems.

Another important concept in promoting inclusivity is *Broadminded*, a sub-value of *Universalism*, defined as someone “willing to accept many different types of behavior, beliefs, or choices in other people” [69]. In software systems, broadmindedness influences the interfaces’ design to become more flexible and inclusive, recognizing user preferences and enabling an acceptance culture where diverse perspectives are respected and valued.

Finally, *Wisdom*, a sub-value of *Universalism*, is defined as “the ability to use your knowledge and experience to make good decisions and judgments” [77]. Wisdom requires Knowledge, as it involves, in software systems, the usage of user feedback and different

data sources to allow the system to make decisions. However, these automated systems require human supervision to ensure everything is working correctly and for sensible choices that the system can't make.

In essence, fostering equality through education, gender equity, accessibility, and the many other concepts represented in this part of the model enables the development of more inclusive and sustainable communities, specifically through the equitable design and implementation of software systems.

4.2.2 Security and Privacy

In this section, we explain the security and privacy part of the conceptual model, presented in Figure 4.3. This part draws on earlier work found during the search process in Chapter 3, having the security part conceptualized in [19] and privacy in [62].

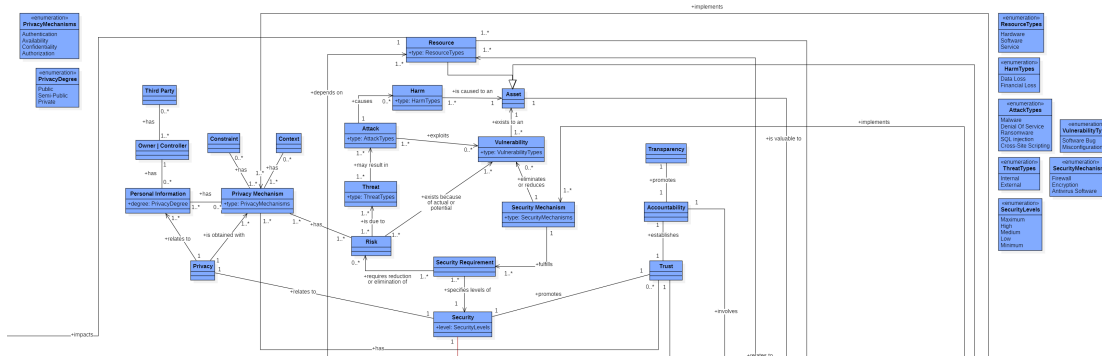


Figure 4.3: Security and Privacy part of the conceptual model.

Security is vital in software systems and serves as an indicator of *Social Sustainability* [2], ensuring all users have safe, equitable access to the system's resources while protecting valuable assets from malicious attacks. Presently, as technology advances, so do cyberattacks. These are becoming increasingly sophisticated, so security is a necessity that systems must implement with diligence to ensure that their information is protected. Failure to properly enforce the required protection would eventually harm the system and its users. Thus, *Security* refers to the measures taken to prevent, minimize, and address threats targeting sensitive assets and corresponds to the level of defense that an asset has against malicious attackers. Depending on how valuable the asset is, *Security* can have different levels. An asset worth a lot will have a higher level of protection when compared to an asset worth less [19].

An *Asset* is something valuable, be it humans or resources (such as hardware or software), and should be protected from possible attacks [19]. *Resource* involves not only the hardware and software but also the functionalities and services provided by the *Software System* that permit stakeholders to access and use these services [2].

Security engineers implement security protocols and technologies, such as firewalls, encryption, antivirus software, and others, corresponding to the concept of *Security*

Mechanism. These mechanisms can be hardware, software, manual procedures, or training, and their objective is to improve security, protect a system from attacks, and ensure that it remains operational. A *Security Requirement* is a quality requirement that defines the level of security needed, concerning system-specific criteria and minimum quality measures, and that a system must maintain to guarantee that it fulfills the security policies. In other words, to protect a system and its assets from threats, security mechanisms are tools or methods implemented to meet the levels of *Security* specified by security requirements, and these requirements outline the protocols and measures needed for the protection [19].

Systems have weak points that attackers may use to attack and get access to the system, causing harm if successful. These flaws are called vulnerabilities, and they endanger an asset. *Vulnerability* examples are software bugs, which are errors or oversights in programs that cause unexpected and unwanted behavior [16], and security misconfiguration, that is, unauthorized access occurs because system configuration settings are missing or wrongly implemented [39]. Consequently, security mechanisms are crucial as they can eliminate or reduce vulnerabilities [19].

An important factor that systems have to acknowledge is the existence of risks in the system. A security risk is the possible damage to an asset, calculated as the “sum (over all relevant threats) of the negative impact of the harm to the asset (i.e., its criticality) multiplied by the likelihood of the harm occurring” [19]. It concerns the unauthorized access or damage to the system and assets. Security risks arise because of threats [19] and vulnerabilities existing in the system [62].

A *Threat* refers to “a general condition, situation, or state (typically corresponding to the motivation of potential attackers) that may result in one or more related attacks” [19]. These threats can be internal or external. An example of internal threats is accidental data breaches due to employees exposing sensitive information by accident and privilege abuse when an authorized user harms the system by installing malware on the network. An external threat can be a distributed denial-of-service (DDoS) attack that overwhelms a network with traffic to make it unavailable to users [105].

An *Attack* is an unauthorized action performed by individuals who want to damage an asset. It may happen by violating the security of a system or bypassing security mechanisms [19]. Examples of attacks include ransomware, denial of service (DoS), malware, SQL injection, cross-site scripting (XSS), and many others [45]. The attack exploits vulnerabilities and can be a success or failure. In case of success, Harm occurs to an Asset [19].

Harm is a “negative impact associated with an asset due to an attack” [19]. It occurs when there is a security violation to an asset and a privacy violation to a user [62]. Potential harm includes data and financial losses. By defining and implementing security requirements, security risks are reduced or eliminated, handling the vulnerabilities of the system, protecting it against potential attacks.

Privacy is an integral part of security. Privacy concerns a person’s right to control personal information, while security involves safeguarding this information. Currently,

a major part of the information is digital and, while this brings advantages in the form of fast and easy access, it also signifies that personal information in digital form must be protected from being revealed to unauthorized individuals [15].

The reality is that most of the time, software systems capture vast amounts of personal information that could end up being used for unintended purposes. To prevent this, privacy must be addressed in the early stages of the software development process [62].

The *Privacy* concept is defined as the “degree to which unauthorized parties are prevented from obtaining sensitive information” [19]. The term “sensitive information” and similar variations, such as personal information, are frequently used in discussions about privacy, but what exactly does sensitive data refer to? *Personal Information* refers to data that identifies or can be used to identify an individual, such as their name, address, phone number, email, or financial details [100]. This information can be private, public, or semi-public and is held by the *Owner / Controller* [62]. This entity has authority over it. The *Owner / Controller* has associations with a *Third Party* [62], a separate entity with an interest or influence in the information.

Personal information requires privacy mechanisms implemented by privacy engineers to safeguard the data from unauthorized access, use, and disclosure [67]. A *Privacy Mechanism* “is a concrete technique to be implemented for helping towards the satisfaction of privacy goals. Some mechanisms can be applied to personal information directly”. Some examples are anonymity, unlinkability, and authentication [24]. These mechanisms are associated with *Constraint*, *Context*, *Risk*, and *Trust* [62].

The *Context* refers to the environment, the software system, where the privacy mechanisms are implemented. On the other hand, a privacy *Constraint* is a “design restriction that is used to realize/satisfy a privacy goal, constraints can be either a privacy policy or privacy mechanism” [24].

Regarding *Risk*, privacy mechanisms are connected to risks, since they must manage the risks that a system needs to consider to ensure the end user’s privacy [62], avoiding unauthorized access, use, and disclosure of personal information [67]. A privacy *Risk* example involves the threat of disclosure a user’s address that could endanger their safety. This is a vulnerability given that a malicious individual could use such data to harm the people living at that address [62]. Thus, privacy concerns can be addressed through various strategies. In this case, a solution might involve implementing data encryption to protect the user’s address from unauthorized access.

Privacy mechanisms offer an additional advantage of building the user’s trust in the system [62]. *Trust* occurs when an individual believes that the system will do what it is supposed to do and won’t misuse the information that it is given [24]. This *Trust* contributes towards the *Community* [63] and is important to establish *Accountability*.

Accountability involves giving information owners the power to hold users responsible for their actions with the information [17]. This principle is assessed using the concept of non-repudiation, which means that users cannot deny their authorship of an action [62]. In software systems, accountability ensures that all actions and transactions can be

traced back to the initiators. This means monitoring user activities, keeping records of all interactions with the system, and thus ensuring that users cannot deny their actions. Furthermore, it involves making sure the system does its duties correctly and registers its actions to prevent and resolve issues.

In turn, *Accountability* promotes *Transparency*. *Transparency* can be defined as “the availability of information about an actor allowing other actors to monitor the workings or performance of this actor” [50]. In other words, it provides clear visibility of system operations and knowledge of who and why an owner’s personal information is collected. This builds a sense of trust as users know how their data is being handled. The concepts of *Trust*, *Accountability*, and *Transparency* were taken from the paper [48].

By addressing both security and privacy concerns, the trust that users place in software systems will increase.

4.2.3 Culture

In this section, we explain the culture part of the conceptual model, presented in Figure 4.4. This part of the model relies on the work presented in paper [40]. While the development of the original model had the organizational culture in the context of ERP implementations in mind, it went through an extension to encompass broader social and online environments because the core concepts of norms, values, and behaviors are universally applicable.

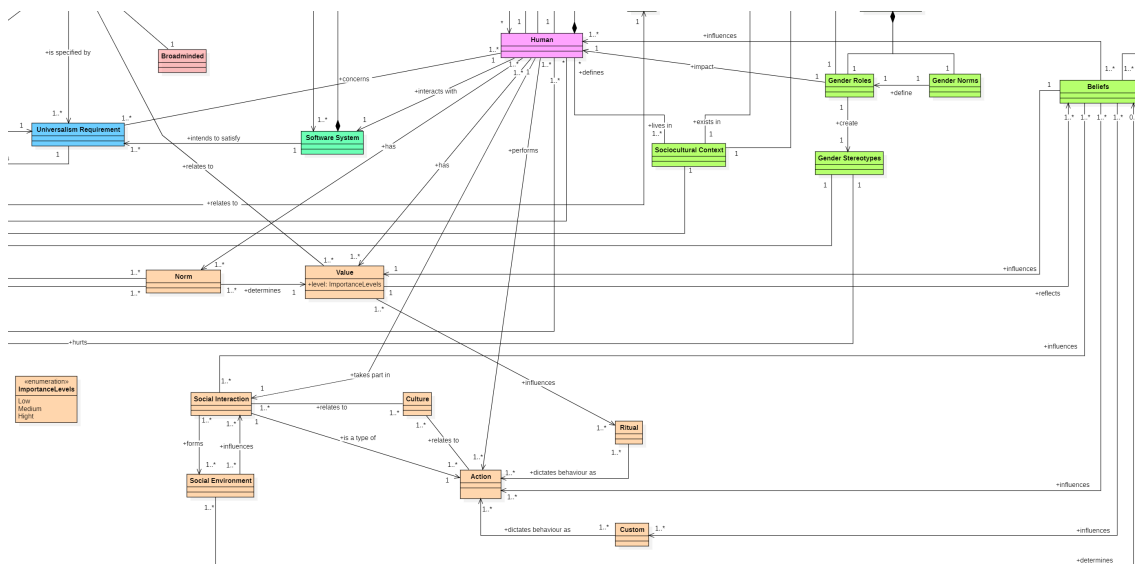


Figure 4.4: Culture part of the conceptual model.

Culture can be defined as "the customary beliefs, social forms, and material traits of a racial, religious, or social group". Additionally, it can also be "the integrated pattern of human knowledge, belief, and behavior that depends upon the capacity for learning and transmitting knowledge to succeeding generations" [51]. In the context of this conceptual model of universalism requirements, *Culture* refers to the behaviors, beliefs, norms, and

values that define how people interact with the system in social and online environments. *Culture* influences the user experience, interactions, and expectations, shaping how users perceive software systems. Cultural factors impact how different user groups view the software and its design. A simple example to better understand this is how culture can affect the design of date and time formats. In the United States, the date format is typically month/day/year (e.g., 08/19/2024), while in many European countries, the format is day/month/year (e.g., 19/08/2024). Adapting the date and time formats to align with local countries prevents confusion for users from different regions.

In this model, *Culture* relates to actions and social interactions because defining norms and expectations will shape and influence how people communicate with each other and use the system. An *Action* consists of “the process of doing something with the intention of achieving a desired goal” [40, 47], that is, in software systems, it’s anything a user does in a system to fulfill a goal. It could be the act of submitting a form or clicking a button, but the essential thing to remember is that actions must be intuitive so that users accomplish their goals effectively in the system.

Actions are culturally influenced behaviors. Cultural context defines how users approach and perceive tasks, impacting the type of actions they prioritize. Let us consider the collectivist and individualist cultures. Countries like China that have a collectivist culture that values conformity to societal norms and alignment with group values [53] may favor systems that highlight teamwork and collaborative features, e.g., Microsoft Teams. On the contrary, users from individualistic cultures like the United States may prefer autonomy, leaning towards tools that permit individual contributions and actions, e.g., personal task management apps. A clearer example of how *Culture* influences *Action* design relates to the fact that in Japan mobile phones must make a sound when taking photographs. This reflects the country’s norms about privacy, where people are dissuaded from taking photos in secret. This *Norm* conditions the design and manner in which users take a photograph. The concept of *Action* in the model relates not only to the functional context but also to cultural expectations.

Humans take part in social interactions. A *Social Interaction* is a type of *Action* that defines an interaction between individuals and can influence their *Beliefs*. The same can be said in reverse, *Beliefs* influence social interactions [40]. “It is a specialization of an action” [40]. In systems, these interactions can occur through messages, content sharing, and participating in discussions on forums. These social interactions will form the *Social Environment* [40], “the immediate physical and social setting in which people live or in which something happens or develops. It includes the culture that the individual was educated or lives in, and the people and institutions with whom they interact” [33]. Putting it in another way, the *Social Environment* is the digital context where users interact with the system and each other. On top of influencing social interactions, the *Social Environment* helps to determine and shape users’ *Beliefs* [40] by providing the context where interactions occur. For instance, interactions on social media will form relationships and support the mutual norms and beliefs in those communities. When users like, share, or comment on

content, they validate the norms and beliefs expressed. Say a post promoting a festival that receives a great amount of likes and comments will reinforce the importance of that festival in the community.

A belief is viewed as a "mental state with properties that can be derived from its representation as a modal operator with possible-worlds semantics" [40, 22]. That is when a person feels certainty about something existing or being true [68]. *Beliefs* influence humans' values, actions, and customs [40].

A *Custom* "is an established behavior expressed as actions, codes, or rules of behavior". Norms, values, and beliefs create and shape customs, which affect "behavior in the form of actions and events" [40]. In other words, customs dictate actions [40] and are established and accepted behaviors that evolve from a culture's norms and values. They shape how individuals are expected to act in different situations. In software systems, customs impact how users behave within the system. They are established user behaviors and practices that develop over time, evolving further when users become familiar with the system. For example, in online forums or question-and-answer websites, it becomes routine for users to 'like' or upvote helpful posts. Eventually, this evolves into an expected action, shaping how users interact and contribute to discussions.

Similar to *Custom*, *Ritual* also dictates *Action* and is influenced by *Value* [40]. Rituals are a group of activities that are "technically superfluous to reach desired ends but that, within a culture, are considered to be socially essential; they are therefore carried out for their own sake" [32]. Therefore, rituals are behaviors that users repeatedly perform, i.e., routine user activities that, while not indispensable for the app, enhance user experience and engagement. For example, daily logins, achievement badges, or interactive challenges are common rituals that promote user retention.

Humans naturally have norms [40], which implies establishing behavior expectations that users must follow. In a system, the concept of *Norm* means that a system must communicate the norms to the users, who must respect them in the community. That is to say, a *Norm* is an expected behavior from a certain group of people in a certain context [5], for example, it is expected of people who find a lost item to return it to the police. In an online environment, most people possess the norm to be respectful while communicating. Additionally, norms influence concepts of *Human Right* and *Fair Salary*, impacting the system policies regarding ethical standards, such as ensuring data privacy as a human right and fair compensation in economy platforms, like Uber.

Every individual has values [40], that is, beliefs connected to emotions and ordered in terms of importance. Values are abstract goals, differentiating them from norms and attitudes, and they serve as standards of criteria [85]. Norms will determine values [40] because norms are expectations within a group in a certain context that will guide an individual's behavior [5]. In time, norms will shape the personal values and priorities of individuals within that group. For example, when in a community there is a prominent norm of supporting and helping others, individuals are more likely to develop values of generosity and sympathy. Values will reflect beliefs [40] by demonstrating them through

their actions. If someone believes honesty is important, they will value integrity exhibited in their actions. While beliefs represent a conviction, values are the principles that stem from those beliefs. In software systems, values are the main goals that influence the system's design and functionalities. These values impact the features and interface to align with users' needs and values. To illustrate, a system may prioritize user privacy, accessibility, or transparency based on its values.

Universalism contains various values, such as equality, social justice, and inclusivity, that are relevant across different contexts. These sub-values help build systems and practices that align with the principles of fairness and inclusivity.

4.2.4 Gender

In this section, we explain the gender part of the conceptual model, presented in Figure 4.5. This part draws on earlier work found during the search process in Chapter 3, having gender conceptualized in [57]. The concept illustration will be done with the car-sharing system [57, 82], which permits users to rent cars using a web or mobile app and charges based on the rental period, distance, and car model. Individuals or companies can supply the cars, and afterward, an identifier is appointed when registered. This identifier will locate the cars on the web and mobile app. Furthermore, it must be parked anywhere in the operational area of the car-sharing service [82].

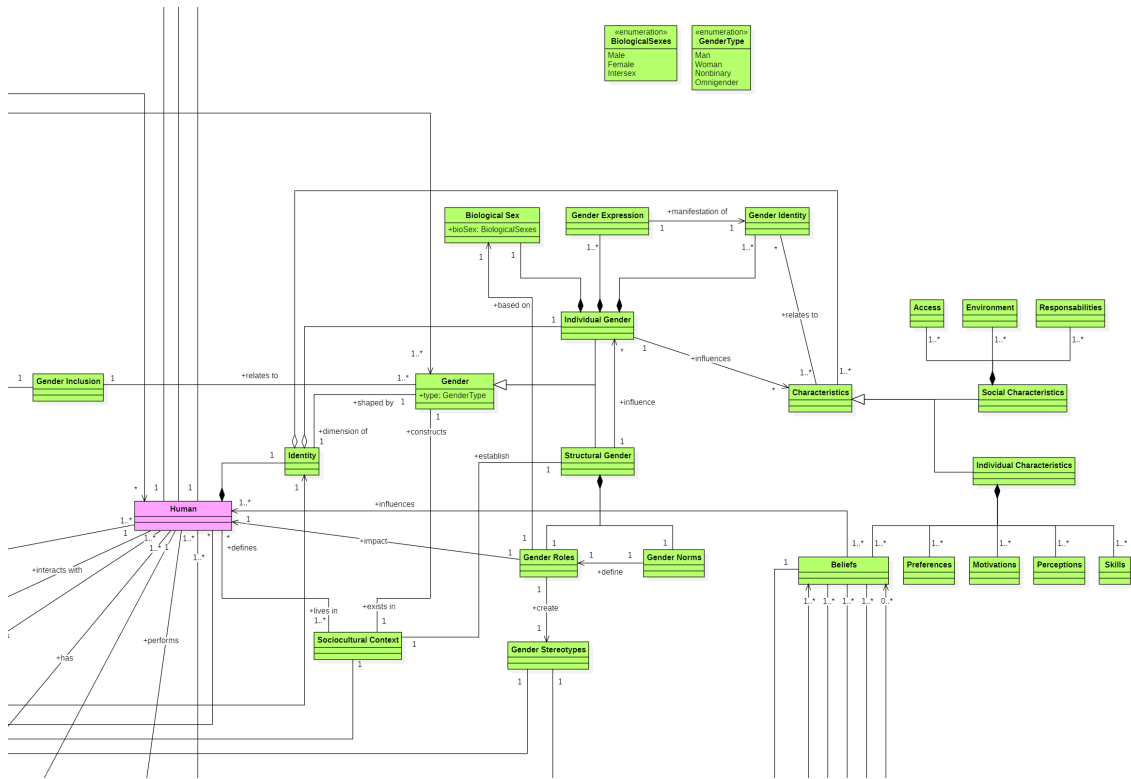


Figure 4.5: Gender part of the conceptual model.

A thriving society must acknowledge the importance of gender inclusion. The issue is that, in many sectors, inequality still prevails, including software development. Since development teams are mainly composed of male individuals, they tend to create systems that favor characteristics common in men, harming other genders. Even if the software is deemed neutral, it often reinforces gender biases and, thus, creates barriers to inclusion. As technology advances and impacts daily life, the haste to address these inequalities so that systems benefit all genders equally increases [57].

Gender Inclusion is essential to fight these inequalities as it promotes *Social Justice* by ensuring that all genders, not restricted only to male and female, are accepted and free from discrimination. In this model, this concept is expanded to include all genders so that software systems support a diverse range of users [98].

Gender is “conceptualized both as a structural feature of society and as a complex part of an individual’s Identity” [57]. *Structural Gender* refers to the societal norms and opportunities assigned to individuals based on their perceived *Biological Sex*, which involves *Gender Roles* and *Gender Norms* [80, 57].

While *Gender* reflects social and cultural constructs, it is strongly linked to *Biological Sex*, which refers to “the biological characteristics that define humans as female or male” [60]. These characteristics, not always mutually exclusive, differentiate individuals as male, female, or intersex [60]. Systems shouldn’t make assumptions or require users to give information about their biological sex unless it’s essential. This promotes privacy and inclusivity allowing users to be safe from feeling categorized or excluded based on their biological characteristics.

Gender Role is “a culturally and socially determined set of expected behaviors, attitudes, and characteristics based on concepts of masculinity and femininity” [9]. In software systems, this means creating features and content that don’t reinforce stereotypical gender roles, allowing users to interact without being constrained by traditional gender expectations. Similarly, *Gender Norms* are “the accepted ideas of how women and men should be, and how they should act, within a specific society or community; they help to define what is understood as being ‘feminine’ and ‘masculine’ within a particular ‘reference group’” [12]. In software, this means creating a system adaptable to diverse cultural and societal expectations, providing adjustable features that align with regional norms.

In the car-sharing example, gender-based division of labor leads to distinct needs and motivations between users of this mobility system. Additionally, *Structural Gender* is influenced by social expectations and interactions within a specific *Sociocultural Context*, which forms imbalances of power between genders [57, 102, 21]. For instance, men may have greater access to certain resources or decision-making roles, which can affect the design and functionality of software systems, resulting in some genders being better served than others. This is how *Gender Stereotypes* arise as generalized views about the attributes or roles men and women are expected to have [97], leading to *Discrimination* and hurting *Gender Equity*. Individuals adopt these stereotypes and perpetuate them through conformity to expected behaviors [57, 10, 37], which can limit their personal

abilities and restrict their choices about careers and life paths [97]. Thus, *Structural Gender* affects *Individual Gender*, a person self-identified gender [80]. In the car-sharing system, this causes variations in daily routines and mobility patterns that correlate with gender [57, 44].

Individual Gender is one part of a person's *Identity* [83], reflecting a human's sense of who they are "as individuals and as members of social groups" [31]. It's defined via their *Gender Identity*, the individual's experience of gender that is separate from their *Biological Sex*, and expressed through their *Gender Expression*. The latter is the outward manifestation of their *Gender Identity* [57, 92], systems must enable users to express their *Gender Identity* in diverse ways, like letting them select from various gender options or enter their own, and choose which pronouns to display to provide a more inclusive representation.

Gender Identity and *Gender Expression* are flexible, allowing individuals to go past the traditional male-female divide and describe how their gender influences behaviors and attitudes in software technologies [57, 92].

The *Sociocultural Context* refers to the social and cultural environment where *Gender* is constructed and the lives of individuals (community members) are defined based on factors such as age, gender, and race. These factors have a big impact on the software development process and the requirements for future systems, affecting the users. In this context, *Gender* is formed and reinforced through the manners in which people learn about and adopt societal norms, as shown by *Structural Gender* [57].

Individual Gender and *Structural Gender* influence the software development process, especially during the requirements phase, where the *Characteristics* and preferences of development teams and stakeholders shape the system's requirements. The fact of the matter is that often the preferences of decision-makers do not mirror the diversity of software users since this context tends to be dominated by a small part of the population, mainly men with higher education and income levels [57, 103, 43].

Software systems are continually developed in environments that overlook diverse contexts under the pretense of neutrality, leading to a reflection of the creators' masculine culture and identity [57, 26, 80]. However, individuals who interact with the Software System are most probably demographically diverse and experience the impact of Gender diversely, leading to different behaviors, preferences, and needs that cause distinct attitudes toward software technologies. Past gender discrimination and the absence of diverse representatives in decision-making limit the perspectives in software development [57].

Disregarding the broader social and cultural factors of software development can accidentally create systems that reinforce existing gender inequalities. This is particularly obvious in the car-sharing system, where diverse gender experiences can be neglected [57]. For example, if the design focuses on late-night availability without considering safety features, like emergency contact options then it may ignore the concerns of women and non-binary users, making the service less accessible and inclusive.

The context where software is developed influences the requirements gathered, so requirements engineers must plan their elicitation processes carefully. Gender issues must

be addressed, like imbalanced power structures that might favor masculine viewpoints, and guarantee the consultation of diverse stakeholders. This implies input from underrepresented groups and using inclusive language by avoiding binary terms. An environment prioritizing gender inclusion will gather the information needed for equitable software solutions [57].

Understanding users' identities and everyday practices is essential for shaping the software's requirements [57]. This will be explored in the section related to the Human part of the model, explaining the *Characteristics* concept as well.

4.2.5 Human

In this section, we explain the human part of the conceptual model, presented in Figure 4.1. *Human* defines the individuals who are involved and interact with the Software System and live in a certain *Sociocultural Context*. *Human* has a unique *Identity* shaped by their *Individual Gender* and *Characteristics*. Humans are influenced by the structural gender systems of their *Sociocultural Context* and contribute to the social construction of gender by incorporating the *Gender Roles* defined by their environment and perceiving others through those same norms. As a result, their *Identity* is shaped by *Gender*, which in turn influences their *Characteristics* [57].

An individual's *Characteristics* stand for their individual and social traits, affected by the user's *Gender*, that define their attitudes and interactions toward software technologies. To facilitate the selection of the relevant *Characteristics* for different *Software Systems*, these *Characteristics* are classified as either individual, reflecting personal attributes, or social, relating to interactions with others [57].

Individual Characteristics include [57]:

- **Beliefs.** Comprised of *Credibility*, *Cost-benefit*, *Self-efficacy*, and *Sense of Belonging*. *Credibility* is the trustworthiness of a software system, *Cost-benefit* is the perceived value relative to the cost of using the software, *Self-efficacy* is trust in the ability to use software in different scenarios, and *Sense of Belonging* is feeling integrated in an online culture or community.
- **Preferences.** Comprised of *Linguistic and Communication Styles* and *Visual Design*. The first is the linguistic and communication styles in an online community, website interface, or software while the second is the appearance of the software interface, like font styles and color.
- **Motivations.** Comprised of *Willingness to learn*, *Motivation*, and *Tinkering*. *Willingness to learn* is the interest in learning more about software, *Motivation* is the factors influencing one's actions toward software, and *Tinkering* is the curiosity behavior when using software.

- **Perceptions.** Comprised of *Risk*, *Financial Cost*, *Ease of Use*, and *Usefulness*. *Risk* is the expectation of possible results from using software, *Financial Cost* is the perception of the financial expense of using software, *Ease of Use* is the expectation that using the technology will require minimal effort, and *Usefulness* is the perception that using the technology will be beneficial.
- **Skills.** Comprised of *Cue Detection*, *Information Processing Style*, and *Awareness*. *Cue Detection* is the recognition of design elements, language, and community standards in a specific software interface, *Information Processing Style* is the approach to understanding new information and addressing challenges in a software task, and *Awareness* is the prior experiences and knowledge related to a software.

Social Characteristics include [57]:

- **Responsibilities.** Comprised of *Time Commitment* and *Routine Integration*. *Time Commitment* is the amount of time available to use a software and *Routine Integration* is how well the software fits with one's habits, behaviors, routines, and environments.
- **Environment.** The *Environment* is the factors that affect how one interacts with the software.
- **Access.** *Access* relates to the access to technological resources.

Gender Identity is conceptualized as a spectrum, therefore an individual may present mixed *Characteristics*, there is no consistency with one gender for all *Characteristics*, influenced by their unique experiences and social contexts. For instance, a person can show a processing style typical among female users while also displaying risk attitudes common in male users, regardless of their gender identity. In the car-sharing system, focusing on *Routine Integration* can emphasize how women balance multiple responsibilities. Engineers who integrate these *Characteristics* in the development process understand the perspectives of numerous users via interviews and creative techniques [57].

This view of the *Human* concept obeys the 'quality of pluralism' from [6], which aims to recognize the complex identities of users in their *Sociocultural Context*. This approach promotes engagement with diversity and challenges the generalized speculations made about users during the early development stages. Utilizing personal attributes that reflect individual identities allows the handling of gender issues without grouping users. This allows for the inclusion of various groups of individuals and acknowledges the different ways they may behave. In the end, to have an effective interaction, the users expect the Software System to be inclusive and accommodate their *Individual Characteristics* and *Social Characteristics* [57].

4.2.6 Software System

In this section, we explain the software system part of the conceptual model, presented in Figure 4.1. The present part is represented by one concept, the *Software System* concept. A *Software System* comprises various resources, such as hardware, software, and services. These are valuable assets that must be protected.

The *Human* actor interacts with the *Software System* that intends to satisfy the *Universalism Requirements*. The knowledge to formulate these requirements must be related to inclusion, accessibility, equitability, and security. Humans expect the design and features of the software system to be safe to use and trustworthy, thus the system must have the necessary security and privacy considerations to safeguard and promote users' trust. Moreover, the software system must offer users the same access and opportunities, without discrimination. In other words, it must not favor gender, race, or other user characteristics. Additionally, the software system must foster a respectful and positive environment where users are safe and abuse-free. It must also make available the necessary resources for people with disabilities so the user base increases.

The software development process produces a *Software System* that bears significant responsibility to its users. It is necessary to explore the *Characteristics* of users that are prioritized and neglected, this will reveal potential biases in the development process. For instance, addressing specific needs, like the concept of *Routine Integration* in the car-sharing system, can help identify specific gender behaviors and improve the general user experience [57].

The system must be designed thoughtfully to ensure it is equally available, usable, and valuable for everyone, promoting inclusivity, accessibility, and fairness across diverse user groups.

4.3 Summary

This chapter presented the conceptual model for universalism values, a start in identifying universalism problems and eliciting the needed information towards the conception of universalism requirements to resolve those issues.

The model was divided into six parts: Equality, Security and Privacy, Culture, Gender, Human, and Software System. Each one had its concepts and relationships explained. First, the Equality part promotes equal access and opportunities for every user in the system, regardless of their background or abilities. Next, the Security and Privacy part of the model advocates the protection of users' data and guarantees their privacy is respected, promoting trust and safety for all users in the system. Then, the Culture part seeks to ensure software systems integrate users' beliefs, values, and norms to improve user experience and encourage positive social interactions. Following, the Gender part ensures the system is inclusive and respects gender diversity, thus adapting to users' gender identities while not discriminating against them. The human part comes after and

represents the individuals involved with a software system. Lastly, the Software System part represents the system that intends to satisfy universalism requirements.

The next chapter will introduce the framework that guides requirements engineers and stakeholders in eliciting universalism requirements for the development of software systems.

A FRAMEWORK FOR ELICITING UNIVERSALISM REQUIREMENTS

In this chapter, we introduce a framework, based on the conceptual model created in Chapter 4, for eliciting universalism requirements. The framework's objective is to incorporate the model into the practices of requirements engineers. It consists of general guidelines, question-based checklists to elicit information for formulating universalism requirements for a software system, and requirements examples to clarify each question. A prototype tool was developed to support the use of the framework. The tool was applied to a case study to demonstrate the framework's practical application.

5.1 Framework foundations

In Chapter 4, the conceptual model presented focused on universalism, addressing potential issues such as equality, inclusivity, and security in software development. This model outlines the essential concepts and relationships that must be regarded when eliciting requirements for systems designed to uphold universalism values. The model is the foundation for integrating these values into software development, particularly during the requirements elicitation activity. To facilitate this integration, we propose the Universalism Requirements Framework to guide requirements engineers in applying the model developed into their practices.

The framework's main objective is to ensure that software systems incorporate inclusivity, justice, and equality by converting each concept of the model into specific software system requirements. The process outcome is a comprehensive document that includes the selected concepts, detailed descriptions, and targeted questions to assist in the requirements elicitation. The framework comprises the practices below to ensure that the values of universalism are seamlessly incorporated into software systems:

- Analysis and study of the conceptual model and taxonomy of universalism values in software development.
- Definition of universalism requirements in the context of software development.

- Plan the elicitation of the requirements that, if respected, will satisfy the universalism goals and make the software safe, inclusive, and equal to everyone using it.
- Deliver explicit and accessible documentation for the addressed concepts.

The framework integrates into the existing practices of organizations or communities. It's not a stand-alone method separated from other software development activities. Instead, it serves as a basis for conversations between requirements engineers and stakeholders. In these discussions, they identify potential universalism challenges in the software system's functionalities to correct and ensure the system is just and inclusive for all users, thus meeting universalism requirements.

Applicability

The framework was developed with the conceptual model as its foundation, so it stands to reason that the model's application scope also defines the boundaries of the framework's applicability. As such, the framework is relevant in software systems whose goal is integrating universalism values, specifically in promoting equality and social justice. In other words, the framework applies to systems where these values are directly or indirectly conceptualized or utilized during the development process. Systems that directly conceptualize universalism values involve software designed to promote social justice and equality through its features and functionalities. An example is applications meant to support underrepresented communities. On the other hand, systems can still impact equality and inclusivity even if not directly focused on universalism values. It all depends on the design. For instance, an online form that does not offer inclusive options for gender or cultural identities limits how a user can represent themselves.

The framework relies on the expectation that the community, organization, or group developing the software will be dedicated to supporting universalism values, such as equality, inclusiveness, and social justice. The framework's success depends on these groups agreeing with, integrating, and maintaining these values throughout the software development process. To apply the framework there needs to be acknowledgment of and confrontation with the inequalities in the software development process that can lead to biased systems. By addressing these issues, the framework seeks to align the software with universalism principles, to promote fairness and justice.

The main users of the framework are the ones who are part of the requirements elicitation process and support universalism values, not individuals or organizations resistant to addressing equality and social justice issues. So, main users include, but not limited to, requirements engineers, software developers, analysts, domain experts, and customers.

5.2 A process to use the Framework

We recommend incorporating a three-step process into the overall requirements elicitation process. These steps are:

Table 5.1: Proposed framework phases.

Pre-elicitation	Create a strategy for the inclusion of universalism values before the elicitation.
Plan the elicitation	Select the knowledge of the model, the techniques and stakeholders
Elicitation process	Elicit information for the formulation of universalism requirements.

5.2.1 Pre-elicitation

The conceptual model highlights that the elicitation of requirements is essential to ensure that universalism values are considered from the very start of the software development process. To successfully elicit these requirements, it is essential to address the four components of the model, specifically Equality, Security and Privacy, Culture, and Gender. Thus, this initial phase focuses on fostering a comprehensive understanding of universalism values and their relations with the software system and its users among all stakeholders involved, so everyone understands the importance of values such as equality and inclusivity.

It's important to establish the right methods to make the requirements elicitation process more effective considering the software's application domain. Methods, such as workshops, interviews, and focus groups, have to be selected carefully based on the context of the software to ensure that diverse perspectives are considered. For example, if the software is a health platform for the community then focus groups may be the better choice, organized with a diverse demographic (e.g., various age groups, socioeconomic backgrounds, and health conditions) to catch many insights and experiences.

The framework's efficacy will depend on the requirements engineers' preparation and comprehension of the model's domain knowledge. Educating requirements engineers and stakeholders on the implications of each value and their contribution to a more equitable user experience will foster the model's understanding. The outcomes of this phase will be work sessions with the managers, requirements engineers, and software engineers to determine the development tasks and techniques aligned with universalism goals.

5.2.2 Plan the elicitation

During this phase, requirements engineers prepare the elicitation process. The document created supports the requirements engineers to integrate the universalism values perspective when planning the elicitation. This document has, for each concept, a description to clarify the concepts, targeted questions designed to gain the stakeholders' perspectives on the concepts, and requirement examples, that is, practical examples that illustrate how the concept translates into requirements. This documentation is a guide to ensure a thorough exploration of universalism values during the elicitation process.

Requirements engineers should study and comprehend the model, this is important to know how universalism values connect with the software to be developed. After this, requirements engineers select and document the universalism concepts that will be addressed during the elicitation. The document is meant to support brainstorming, creativity techniques, interviews, or questionnaires, having available a proper space to note down the stakeholder's answers regarding the topic discussed.

5.2.3 Elicitation process

After finalizing the planning, requirements engineers will communicate with stakeholders to collect data concerning the identified universalism values. This involves using the prepared documentation as a reference during interactions with stakeholders to facilitate discussions about each concept.

Requirements engineers must serve as facilitators so that conversations are open and inclusive, to permit every perspective to be considered. The questions will guide the discussions, helping to identify potential issues and insights related to universalism values in the software system. The outcome of this phase will be a set of insights and requirements that reflect the stakeholders' perspectives on integrating universalism values into the software.

5.3 A Supporting Tool for the Framework

A web-based prototype tool was developed to support the elicitation planning phase of the framework. The tool has a clear workflow path with three intuitive steps to create the universalism requirements document, which is the result of the process. The three main steps are:

- **Concept Selection:** The user has an interactive model where he can choose the concepts to select after being given a description.
- **Question Selection:** A list of questions is present for the previously selected concepts, and the user chooses the questions they want to address.

- **Document Creation:** The user submits all the details for the requirements document, and a file is created and exportable for the elicitation process later.

5.3.1 Prototype development

We implemented the prototype ¹ in React, a JavaScript library for building user interfaces. The landing page for the prototype tool is in Figure 5.1. It has three buttons displayed. The “What is this tool?” button leads the user to a page that explains the purpose of the tool, to better understand why this tool was created while the “See instructions” button leads to a page that explains how to use the tool. A user familiar with the tool can open it directly and start the process with the “Start” button. The navigation bar is always the same. Clicking “Home” redirects the user to the home page and “Documents” to the documents page, which has a list of all the documents of the user.

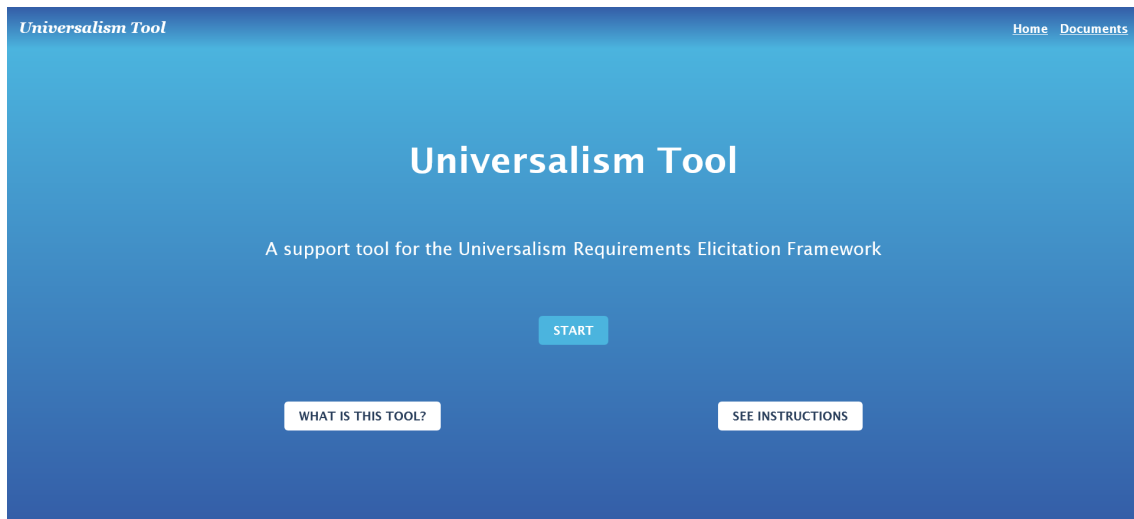


Figure 5.1: Landing page of Universalism tool.

Figures 5.2 and 5.3 show the first step in the process. The conceptual model is in the center of the page, and inside the blue rectangle, it's possible to navigate and zoom in and out to see and explore the model and its concepts. A mini-map is available on the bottom right corner of the rectangle so that users can situate the part of the model they are focusing on within the whole conceptual model. Due to the size of the model, the user can isolate the sections of the model that they desire. For this, the user just has to select in the blue bar, located under “Model Sections:”, the section they desire, and it will appear inside the rectangle.

The characteristics of a concept only appear after the user clicks on it. A table will appear below the blue rectangle with a description of the concept to better understand a concept. More abstract concepts, such as wisdom, will have an additional phrase to explain what it would look like in a software system. After reading, if the user decides to

¹Universalism Tool

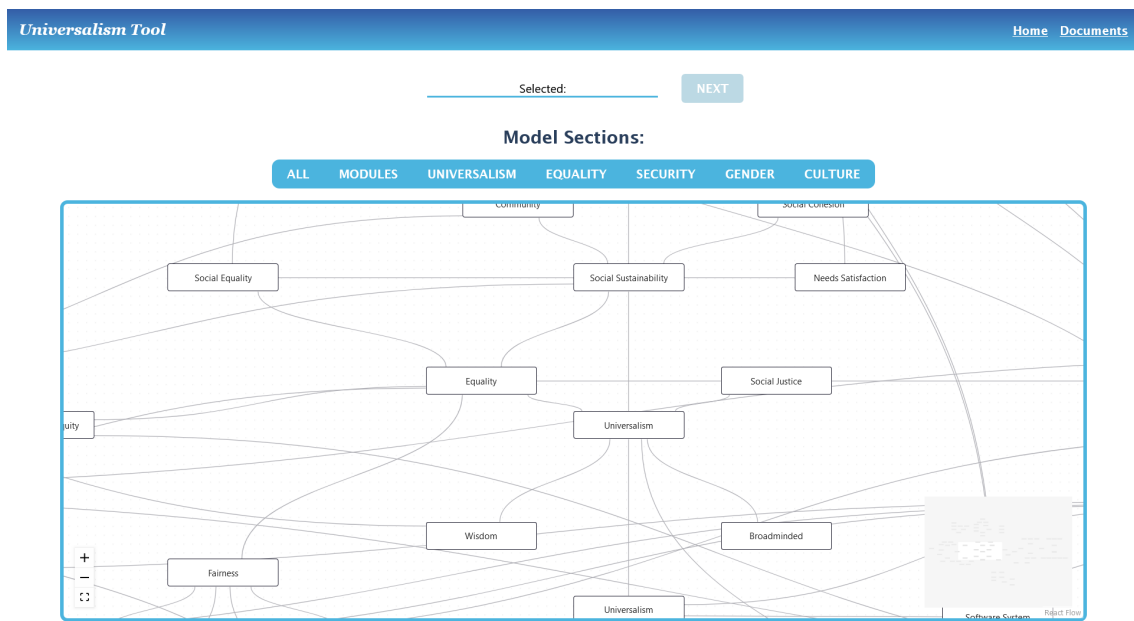


Figure 5.2: Concept selection from the model.

add the concept to the final document, he has to click the “Add to document” button on the top right corner of the table. Then, a pop-up will appear to confirm. A concept can only be saved once, and if the user tries to add the same concept more times, a pop-up will appear to inform that it was already saved. The user can add all the concepts he wants as there are no limits. When the user wants to continue the process, they click the “Next” button to redirect to the questions page.

Equality		ADD TO DOCUMENT
Description	Equality is one of the sub-values of Universalism. It's defined as the right of different groups of people to have a similar social position and receive the same treatment. In software systems, equality presents itself when all users have the same access and opportunities, without discrimination. This includes applications that do not favor gender, race, or other user characteristics. Equality also promotes fairness in the workplace through policies and practices.	

Figure 5.3: Characteristics of the selected concept.

Figure 5.4 shows the process of selecting questions for the previously selected concepts. At the top are the “Previous” and “Next” buttons, in case the user needs to go back to add concepts. If the user wants to eliminate a concept, he can remove it from the list by clicking the “X” next to the concept, or simply not selecting any question will exclude the concept from the final document. Each concept has a set of questions, and only the concepts and questions selected will appear in the final document.

Additionally, every question has under it a requirement example in italics, taken from an existing application: ShareTheMeal. If any of these questions do not directly apply

CHAPTER 5. A FRAMEWORK FOR ELICITING UNIVERSALISM REQUIREMENTS

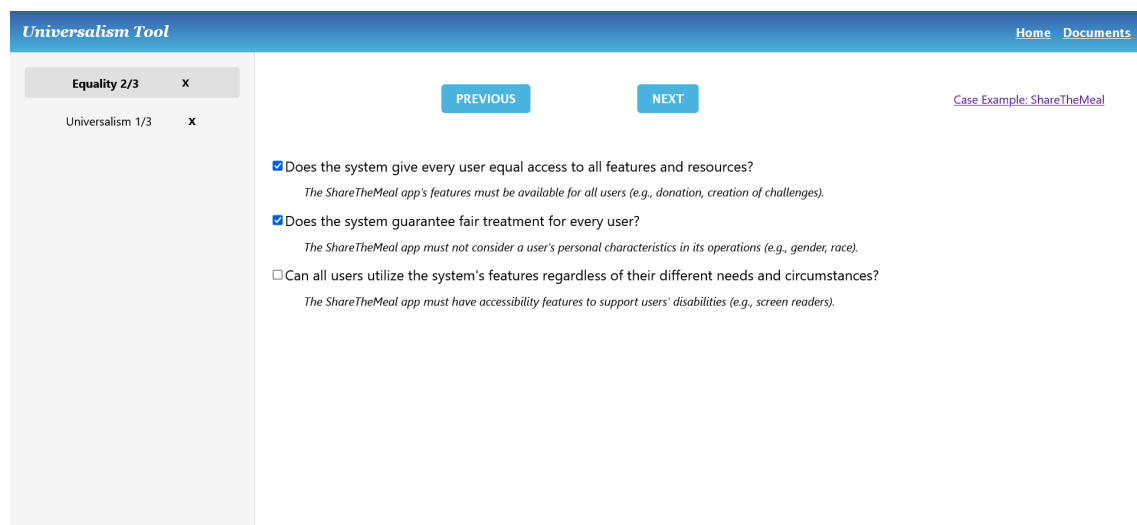


Figure 5.4: Questions' selection of the selected concepts.

to the ShareTheMeal app, then we bend or extend the example to fit the concept and questions. To learn more about the application, the user can click on the link in the top right corner of the page, Figure 5.5, which leads to a different page where the user can find a brief description of the app.

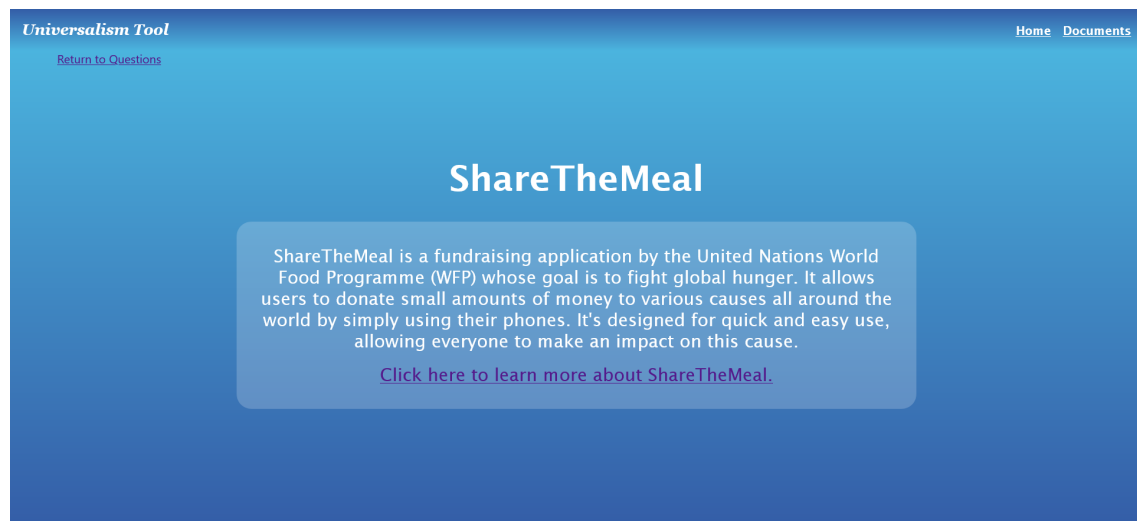
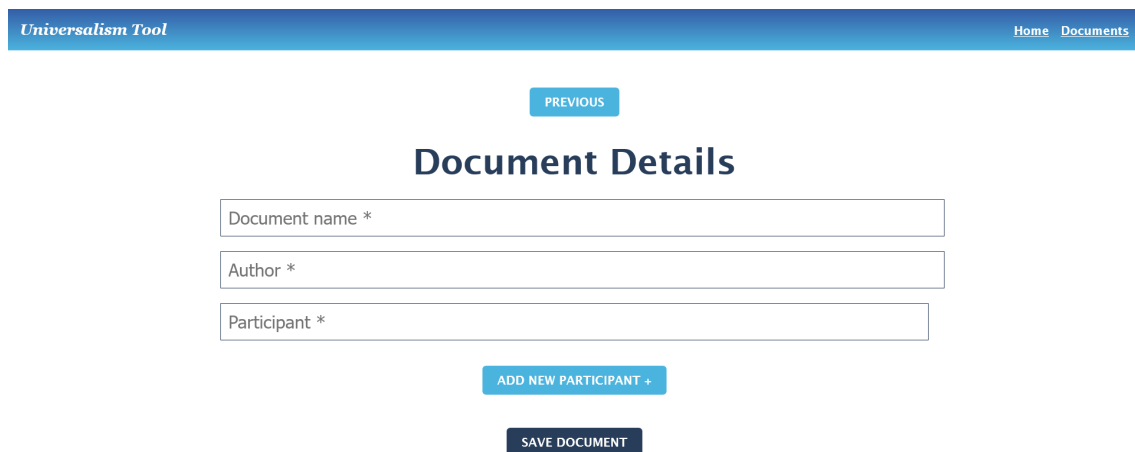


Figure 5.5: ShareTheMeal app description.

Next comes the document page, Figure 5.6, where the user can give the document a name and add the name of the author of the document along with the participants. The user can add as many participants as he wants by clicking the "Add new participant +" button. These are the final details asked to create the requirements document. After doing so, the user can click "Save document" button to save the document, he will be redirected to the documents page, where the user can see all the documents created.

The documents page, Figure 5.7, provides a list of all documents. Each document



Universalism Tool [Home](#) [Documents](#)

[PREVIOUS](#)

Document Details

Document name *

Author *

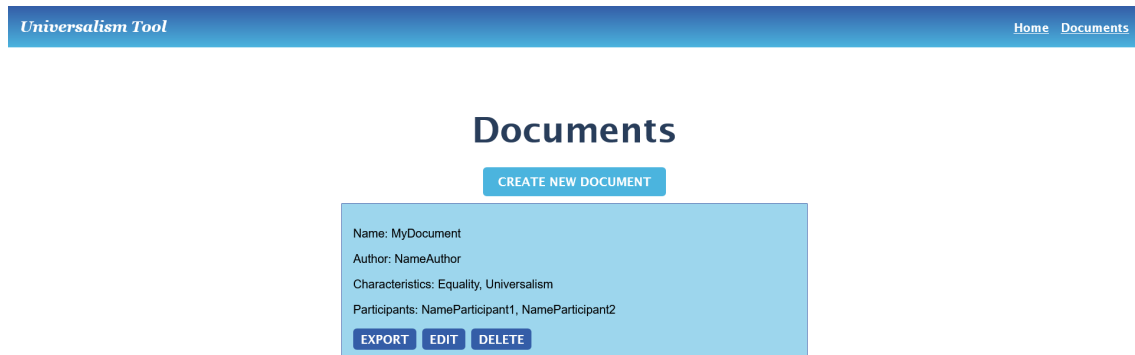
Participant *

[ADD NEW PARTICIPANT +](#)

[SAVE DOCUMENT](#)

Figure 5.6: Document page with the form for the details.

presents three buttons: “Export”, “Edit”, and “Delete”. As the name indicates, the “Export” button will export the user’s document in a docx format file. The “Edit” button will edit the chosen document. And the “Delete” will erase the document from the list. The “Create new document” button will redirect the user to the tool page where he can start the creation of his new document.



Universalism Tool [Home](#) [Documents](#)

Documents

[CREATE NEW DOCUMENT](#)

Name: MyDocument
Author: NameAuthor
Characteristics: Equality, Universalism
Participants: NameParticipant1, NameParticipant2

[EXPORT](#) [EDIT](#) [DELETE](#)

Figure 5.7: Documents page with the list of documents.

Figure 5.8 is the resulting document from the export functionality, with the author and participants’ names, the concepts selected with the related description, and the questions selected. Each question has a designated space for the answers of each participant.

Universalism Requirements Document

Project Name: MyDocument

Author: NameAuthor

Participants:

- NameParticipant1
- NameParticipant2

Selected Concepts:

Equality:

Description:

Equality is one of the sub-values of Universalism. It's defined as the right of different groups of people to have a similar social position and receive the same treatment.

In software systems, equality presents itself when all users have the same access and opportunities, without discrimination. This includes applications that do not favor gender, race, or other user characteristics. Equality also promotes fairness in the workplace through policies and practices.

Question:

Does the system give every user equal access to all features and resources?

Answer NameParticipant1:

Answer NameParticipant2:

Question:

Does the system guarantee fair treatment for every user?

Answer NameParticipant1:

Answer NameParticipant2:

Figure 5.8: The final requirements document.

5.4 Applying the Framework to a Case Study

In this section, we applied the framework to a case study to demonstrate the use of the conceptual model and the prototype tool developed. The one selected was the website-based application Freerice, an educational trivia game that helps users gain knowledge while at the same time causing a positive impact on various communities in the world. For every question the user answers correctly, 10 grains of rice are collected for the World Food Programme (WFP) to support its mission of saving and improving lives worldwide. In this application, rice is simply a metaphor to make the game palpable for players as every correct question generates the equivalent of ten grains of rice as payment to the United Nations World Food Programme's operations globally. The money collected comes from private sponsors that match the rice grain donations generated in the application [78].

Due to its alignment with universalism values, particularly equality and social justice, Freerice was selected as a case study. The application effectively teaches users knowledge of diverse categories and, at the same time, takes advantage of that to aid those in need, helping on two fronts.

The web application itself possesses many strong points. The game is simple to understand, does not require a user account to contribute, and offers many different question categories, such as English and History, with distinct difficulty levels. The game includes challenges to increase user engagement with the app. These are open to everyone, but it is possible to create groups with people the user knows to host competitions to collect rice grains. Users can track their progress through individual and group leaderboards, adding an element of competition to motivate users [78].

The game is accessible via web browsers on smartphones, tablets, and computers. The Freerice creators decided to focus their resources on the responsive web app, but wish to reintroduce the mobile apps in the future [78]. Freerice is a great contributor to equality and social justice, thus a great example to apply our framework. So, with this in mind, we will use the framework to generate a document with the necessary questions to address or note during the elicitation process.

Starting with the pre-elicitation step, we analyze the domain of the Freerice app and identify the values that must be addressed before its planning and design. Given the game's intention, it is evident that the *Equality* component of our model is essential and should be a primary consideration. Accessibility and compatibility are crucial elements to ensure that the app reaches as many people as possible, thus maximizing its impact in collecting donations. This includes considering users with disabilities and varying technological capabilities, which helps to expand the pool of potential users. So, the concepts selected are *Disability* and *Compatibility*. The main mission of the app is to use learning as a tool to generate contributions as rice grains, making the concepts of *Education* and *Knowledge* vital to its design. Additionally, to encourage user engagement and increase motivation, it is equally important to build a sense of community among users, hence

CHAPTER 5. A FRAMEWORK FOR ELICITING UNIVERSALISM REQUIREMENTS

selecting the *Community* concept. This aspect improves the experience and encourages collaboration.

Concerning the *Cultural* part, three concepts stand out: *Culture*, *Norm*, and *Social Environment*. The app's design and content must align with users' diverse backgrounds, and by integrating cultural expectations, it can create a meaningful user experience that adapts educational content and challenges to reflect the user's region. This is an effective strategy to make more users interested in the app. The concept of *Norm* is essential for establishing clear behavioral expectations within the app's community, striving for a respectful and supportive environment. Finally, the *Social Environment* element focuses on creating a digital space where users can connect, share experiences, and collaborate. These interactions will further encourage user engagement and build a sense of community.

Regarding the *Security and Privacy* part of the model, the following concepts stick out: *Privacy*, *Privacy Mechanism*, *Transparency*, and *Security Mechanism*. *Privacy* is vital since protecting users' personal information promotes trust and encourages participation in the app. To enhance this trust, *Privacy Mechanism* is an essential concept for users to have control over their personal information through features that allow users to manage their privacy settings and understand how their data is processed. Additionally, implementing *Security Mechanisms* is necessary to protect user data from potential breaches. Also, *Transparency* builds the confidence of users using the app, and, for this, they must be informed about how their data is used and the benefits of sharing their information.

In this game, gender is not directly included as an explicit feature or requirement. The app does not need to ask for or use the user's gender to interact with the game. Rather, it is considered indirectly, meaning the app recognizes that it can influence how users interact with software but doesn't require a user's gender to function. Therefore, the app doesn't need to consider each individual's gender separately, it focuses on broader characteristics that may vary by gender. So, for the *Gender* part of the model, the concepts selected are *Linguistic & Communication Style*, *Motivation*, and *Willingness to Learn*. The first indicates that the app must support multiple languages and accommodate various communication styles to allow users from diverse backgrounds to fully engage with the app's features. *Motivation* is what drives the user to use the app and *Willingness to Learn* is vital for user engagement and retention. The Freerice app should provide educational resources and incentives that encourage users to explore its functionalities.

After listing the values we want to explore, the planning of the requirements elicitation is the next step, where the document to support the final step, the elicitation process, is created. The second step is where we will employ the tool, the prototype developed and presented earlier. Through the use of this tool, we add the concepts relevant to the creation of the app, and then we select the questions from each concept that we are interested in.

Let's use the *Disability* concept as an example. In the tool, we select the *Disability* concept in the model presented on the page, after that a description will appear, explaining that there are many types of disabilities to consider and that the system must accommodate. After saving the concept, we go to the next page with the questions, where we analyze the

questions and select the ones we want.

- **Does the system accommodate users with disabilities (e.g., physical, neurological, cognitive, auditory, visual, and speech)?** The app must be compatible with screen readers, allowing keyboard navigation, providing alternative text for images, using clear fonts and colors, and offering simplified language options for users with cognitive disabilities.
- **Does the system have customizable settings for the specific needs of users with disabilities?** The app should allow users to personalize their experience by adjusting text size or modify color contrast. Another feature the app must have is speech-to-text and text-to-speech, so users can respond to the questions, create groups, and participate in challenges.
- **Are users with disabilities involved in the testing and feedback process?** People with disabilities should be involved in the app's development process. This means creating focus groups for user testing and using dedicated channels for feedback.

Finishing with the selection of the knowledge of the model, we proceed to choose the right techniques and stakeholders for the elicitation process. We identified two groups as our main stakeholders: everyday users, including individuals with disabilities, and the World Food Programme (WFP).

To collect these stakeholders' perspectives, we chose two different techniques that we found most suitable:

- **Focus Groups.** This technique will be utilized with users. Focus groups facilitate open discussions among the groups, to explore potential challenges and preferences they may have about the app. This way, we capture diverse perspectives to make the app more accessible and engaging for all users.
- **Interviews.** Closed interviews will be conducted with representatives from the World Food Programme. This face-to-face conversation will help us gain insights into how the app can align with their mission, current needs, and priorities.

With the three necessary elements selected, i.e., knowledge of the model, the techniques, and stakeholders, we move to the page following the questions page, the document page. Here, we fill in the details regarding the document and save it. Next, we use the export feature to have our requirements document, presented in Figure 5.9.

In the end, we have composed a document with the necessary questions to address during the elicitation process.

Universalism Requirements Document

Project Name: Freerice App

Author: Requirements Engineer A

Participants:

- User B
- WFP Representative C

Selected Concepts:

Disability:

Description:

Disability is an illness, injury, or condition that makes it difficult for someone to do some things that other people do, and that is usually permanent or lasts for a long time. There are many types of disabilities, they can be Physical, Neurological, Cognitive, Auditory, Visual, and Speech related. Software systems must accommodate users with different disabilities.

Question:

Does the system accommodate users with disabilities (e.g., physical, neurological, cognitive, auditory, visual, and speech)?

Answer User B:

Answer WFP Representative C:

Question:

Does the system have customizable settings for the specific needs of users with disabilities?

Answer User B:

Answer WFP Representative C:

Question:

Are users with disabilities involved in the testing and feedback process?

Answer User B:

Figure 5.9: The Freerice requirements document.

5.5 Summary

This chapter introduced a framework to integrate the conceptual model into requirements elicitation techniques. First, we present the basis for the construction of the framework, followed by a description of the process of using it. Afterward, we showed the development of a prototype tool to support the process. In the end, we applied the framework to a case study.

EVALUATION

In this chapter, the evaluation of the conceptual model for universalism requirements is presented. For the evaluation, static validation was utilized where the proposed model was introduced and complemented with a video presentation of the prototype tool to the participants. They analyzed it and gave their opinions regarding the work this dissertation presented by filling out a questionnaire. This chapter explains the questionnaire form created, followed by the survey results. Finally, the threats to the validity of the evaluation are discussed.

6.1 Evaluation planning

The evaluation is conducted to gather different perspectives, from novices to experts, about the conceptual model for universalism requirements. With this in mind, both quantitative and qualitative data were collected to assemble a comprehensive understanding of the thoughts on the model and their reasoning. These types of data are collected together using an online form. The form included a questionnaire about the model and the prototype tool. It also had links to access the tool, a guide to the support tool, and, alternatively, a video explaining how to use the tool. Another material provided was the framework document that complemented the understanding of the model and the tool, this one was optional.

The questionnaire was made with Google Forms, a web application that permits the creation and management of online surveys, for free. Google Forms has a feature that exports the questions and answers of the questionnaire into a spreadsheet. The video and documents were uploaded to Google Drive for sharing through links. The section below describes the form created and the guides.

6.1.1 Questionnaire

The questionnaire ¹ consists of quantitative questions, e.g., multiple choice and Likert scales, and qualitative questions, e.g., open-ended. It was written in English but the participants

¹An Approach For Human Values Requirements: The Case Of Universalism Questionnaire

could answer in English or Portuguese. A PDF version of the questionnaire is shown in Appendix A. The form was organized into 11 sections that will be explained next.

- **Introduction.** This section informs the participants of the context and purpose of the evaluation. It provides all the necessary links for the participant to access and view the guides and prototype tool. It also reports that the answers are anonymous and how long it takes to answer the questionnaire. The contact information for any participant who wants to follow up on the results of this work is available.
- **Pre-survey.** This section collects the participants' personal information about their gender, age, academic training, scientific area of the degree, current occupation, years of experience with conceptual modeling, and the importance they give to universalism values during software development.
- **Overview of the Model.** This section presents an image, a representation of the model and its various parts, and a brief description of the main goal. It also offers a link for the participant to access the full model, which, due to the size, is not feasible to present directly in the form.
- **Components.** Six sections were created for each part of the conceptual model: Security and Privacy, Equality, Culture, Gender, Software System, and Human. The sections include a Likert scale question that asks if the model part has all the necessary concepts and relationships. Then, there is an open-ended question that prompts the participants to explain what they would change in the model.
- **General opinion about the conceptual model.** This section gives the participants the link for the complete model and requests their general opinion about it. For this, it presents 4 Likert scale questions and 2 open-ended questions. The first four questions relate to the model's usefulness, the participants' difficulty in understanding it, their willingness to use the model to develop an application, and whether they would recommend it. The open-ended questions ask the participants their opinion on the strongest and weakest points of the model.
- **Prototype Evaluation.** This section gives the links to access the prototype tool, the document guide, and a video on how to use the tool. The participants are advised to read the guide or watch the video before answering. This section was divided into questions related to the diverse tool pages, Likert scale questions were used and open-ended to understand issues the participants might have had. In the end, there were questions about the usefulness of the framework document, the guide document, and the video.

6.1.2 Prototype Guides

Two different guides were created to introduce the participants to the prototype tool that implements the framework. The first guide² was a Word document that introduced the context of the tool, and then provided a written description for each page of the app, with all the necessary prints of the tool, explaining the entire process. The second guide³ was a video with step-by-step instructions and a voice-over, it had a total of 3:34 minutes. Both guides were in English.

6.1.3 Framework Document

The framework document⁴ complements the understanding of the model and the tool. It is composed of a deeper description of the requirement example, the ShareTheMeal application, and all the concepts of the model, separated into the different parts of the model. For each concept, there is a description, the questions, and requirements examples. This document presents the references used.

6.2 Results and discussion

The discussion of the results followed the order from the questionnaire. First, the general results from each question are presented and discussed, followed by a more detailed analysis incorporating the participants' qualitative responses to complement and enhance the quantitative analysis.

The form responses were gathered for ten days, resulting in 24 submissions. All participants had some training and experience in the computer science field, which will be examined later in this chapter.

6.2.1 Participants demographics

These questions gave information about the characteristics of the participants and their perception of universalism values.

In terms of gender, the 24 participants showed a male majority, with 15 men (62.5%), 8 women (33.3%), and one person who preferred not to disclose this information (4.2%), as shown in Figure 6.1.

Regarding the 24 participants' age, as Figure 6.2 presents, they ranged from 22 to 60 years, showing a diverse age distribution. However, mainly the participants fell in the 23 to 25 age range, with 5 (20.8%) participants aged 23, 7 (29.2%) participants aged 24, and 4 (16.7%) participants aged 25. The remaining individuals were spread across ages 22, 27, 30, 33, 37, 41, 53, and 60 (4.2%).

²Guide for the Universalism Tool

³Universalism Tool Video

⁴Universalism Requirement Framework

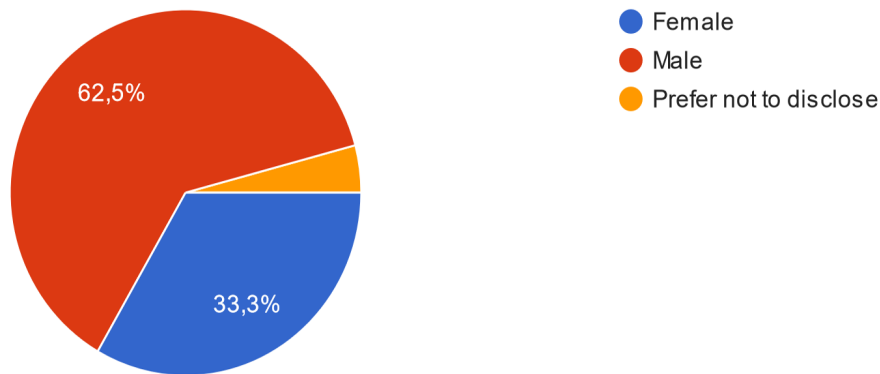


Figure 6.1: Percentage of participants by gender.

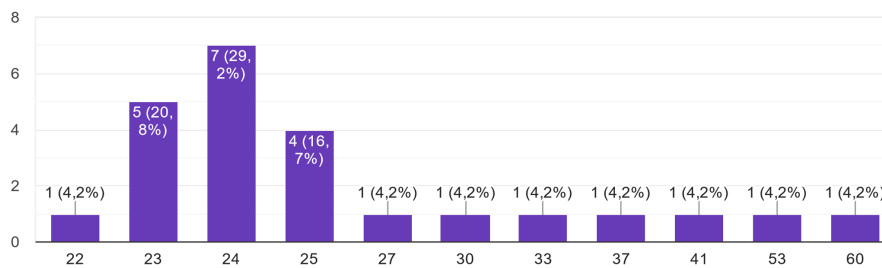


Figure 6.2: Percentage of participants by age.

The participants' academic training was diverse, 15 (62.5%) participants had completed their Bachelor's, 5 (20.8%) had their Master's Degree, and 4 (16.7%) had their PhD. Figure 6.3 shows the results.

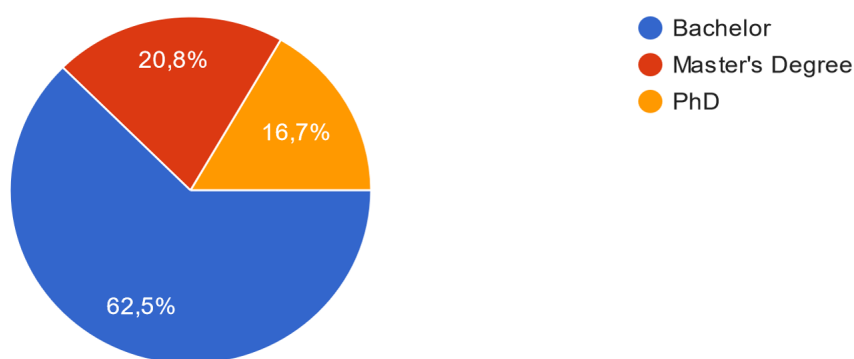


Figure 6.3: Percentage of participants by academic training.

The scientific area of the participants' degree wasn't diverse, Figure 6.4, as 22 (91.7%) participants said computer science, 1 (4.2%) distributed systems, and 1 (4.2%) software engineering.

Then, the participants were asked about their current occupations and, as shown in Figure 6.5, out of the 24 participants, 17 (70.8%) were students, 4 (16.7%) practitioners, 2

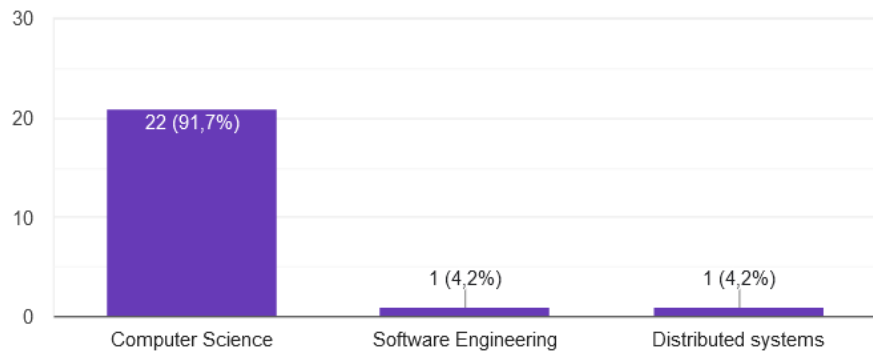


Figure 6.4: Percentage of participants by scientific area of the degree.

(8.3%) academics, and 1 (4.2%) researcher.

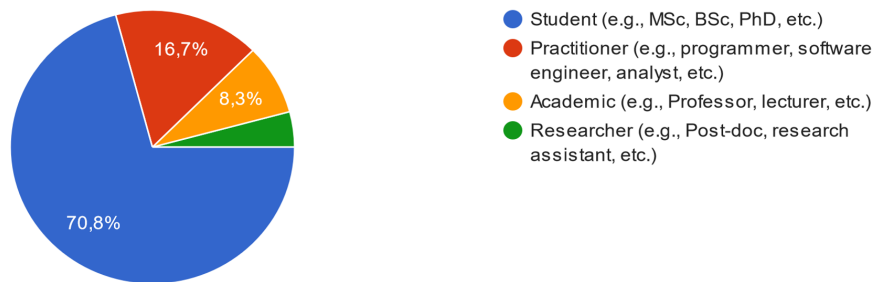


Figure 6.5: Percentage of participants by current occupation.

Figure 6.6 presents a total of 16 (66.7%) participants who had less than two years of experience in conceptual modeling, 5 (20.8%) had three to five years of experience, and 3 (12.5%) had more than ten years of experience. These results show that most of the participants had little experience in this area of conceptual modeling.

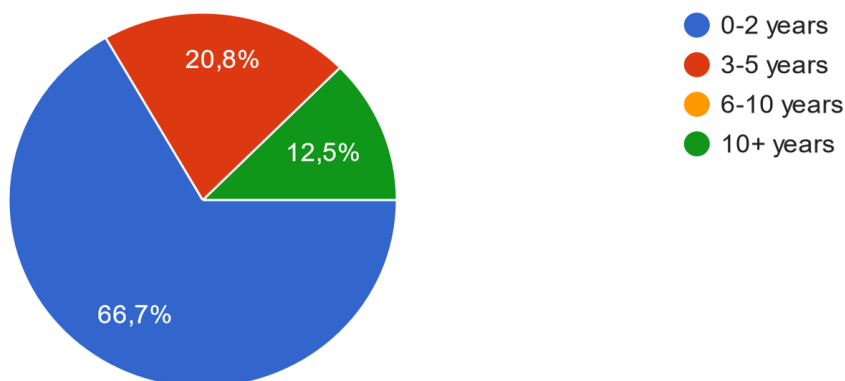


Figure 6.6: Percentage of participants by years of experience.

Perceptions of universalism values

This analysis was to comprehend participants' opinions on the importance of universalism values—such as equality and social justice—during software development. These perspectives are essential for assessing whether participants recognize the significance of these values in the development process, or whether they view them as less relevant, potentially reflecting on the perceived value of the model itself.

So, participants had to rate their opinion on the importance of universalism values in software development on a scale from "1" (Not Important) to "5" (Very Important). As shown in Figure 6.7, 10 (41.7%) participants rated this topic as very important, 7 (29.2%) considered it important, 4 (16.7%) viewed it as moderately important, 1 (4.2%) indicated that it was slightly important, and 2 (8.3%) did not consider it important at all.

Despite the variety of results, it's clear that most participants believe that universalism values play an important role in software development, showing that the model's context is relevant.

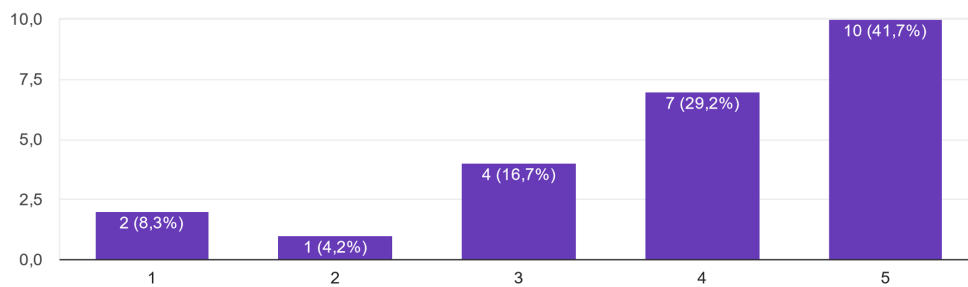


Figure 6.7: In your opinion, how important are universalism values - such as equality, social justice - during software development?

6.2.2 Components

In this section, the results from each part of the model's evaluation are presented. Each consisted of the same two questions. The first asked whether the model part in question included all the necessary concepts and relations, this was a Likert scale question where 1 means that the participant strongly disagrees and 5 means that he strongly agrees. The second was an optional open-ended question where participants were encouraged to express what they would change or add about the model part in question.

6.2.2.1 Security and Privacy

The evaluation of the Security and Privacy part of the model produced positive results. As demonstrated in the first question (Figure 6.8), which asked if the model properly included all necessary concepts and relations regarding Security and Privacy, most participants agreed that the model was well-constructed. Specifically, 8 (33.3%) participants rated it a

5 (Strongly Agree), 12 (50%) rated it a 4, 3 (12.5%) gave it a 3, and only 1 (4.2%) rated it a 2. No one rated it a 1, indicating that most found the model satisfactory.

However, the open-ended answers to the second question provided valuable insights for improvement. The 5 responses highlighted areas where additional concepts or refinements could improve the model.

One participant suggested: "I think some concepts related to patch management and recovery might be missing, secure systems should be able to roll-back or restore secure states, which should be stated concretely in the diagram.", which is an important aspect of maintaining security, especially after a breach or failure.

Another three participants proposed additions to the model, such as, including data leaks as a potential HarmType from a privacy viewpoint, implementing security mechanisms like "Input Validation and Sanitization," and referencing ISO27k:2022 attributes like identify, protect, detect, respond, and recover. Furthermore, one participant suggested that additional resource types (e.g., data), attack types (e.g., phishing), and security mechanisms (e.g., access control) could also be included.

One insightful comment noted: "hard to say, but each of these classes may have different interpretations, and some relations may also not be the best choices. For instance, making security to promote trust, is just a partial view.". This meant that some relations between classes may not be ideal, since linking security to trust was seen as "a partial view," which suggests that there needs to be consideration for other security goals beyond trust.

These recommendations show that the Security and Privacy part of the model needs further refinement and expansion. Even though the general structure and concepts are well-received, incorporating these elements will strengthen the model and make it more relevant to real-world systems.

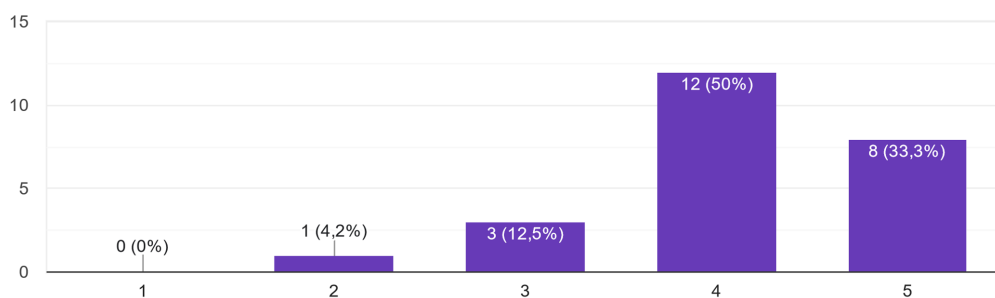


Figure 6.8: Does this model properly include all the necessary concepts and relations regarding Security and Privacy?

6.2.2.2 Equality

The evaluation of the Equality part of the model produced the same positive feedback. As the first question (Figure 6.9) shows, it asked if the model properly includes all necessary concepts and relations regarding Equality, 11 (45.8%) participants rated it a 5 (Strongly

Agree), 10 (41.7%) rated it a 4, 2 (8.3%) gave it a 3, and 1 (4.2%) rated it a 2, with no participants rating it a 1. This indicates that, overall, the model captures the essential elements of Equality.

Once again the open-ended responses gave vital insights to refine the model. One participant said: "The relationship of discrimination and fairness should not be "is an indicator of", if discrimination exists then fairness does not". This comment helped the realization that there needs to be a clearer representation of the *Discrimination* concept so that its purpose of addressing equal opportunities is evident. This feedback spots a concept that needs refinement in the model.

Another participant mentioned the potential benefit of referencing consent forms and terms and conditions, emphasizing the importance of clear and accessible information for users.

A third comment questioned: "I have empirical knowledge in this area, so my contribution is not as in-depth as I would like. Still, I wonder what is the usefulness of having "SafetyFeatures" in FairTreatment?". In response to the participant's question, it's important to explain that *Fair Treatment* ensures equal treatment for all individuals, which includes fair evaluations and opportunities based on merit. In software systems, this also means designing features that mitigate personal liability in stressful situations. Safety features, such as error handling, prevent users from being penalized for accidental mistakes, and data validation reduces errors before they happen, promoting a supportive environment. Thus, safety features align with the creation of equitable systems. This response, however, shows that further clarification may be needed to address any uncertainties.

Lastly, a comment raised concerns about the clarity of certain relationships within the model. It questioned the implications of *Fair Salary* and *Education* regarding the goal of promoting equal access. The participant suggested that the model should have a wide scope to consider factors such as software availability, public awareness, and cost to achieve equitable access.

Overall, these insights indicated the strengths and weaknesses in the Equality part of the model. While, in general, the answers show that the model effectively captures the concepts related to Equality, the feedback indicates that there must be clearer definitions and relationships. Addressing these comments will improve the model's strength in reflecting the complexities of equality in software systems.

6.2.2.3 Culture

The evaluation of the Culture part of the model provided a mixed set of results. According to the first question (Figure 6.10), 7 (29.2%) participants rated it a 5 (Strongly Agree), 11 (45.8%) rated it a 4, 4 (16.7%) gave it a 3, while 1 (4.2%) rated it a 2, and another 1 (4.2%) rated it a 1. This implies that while many participants agreed with the cultural elements presented, there remains room for improvement and refinement.

The open-ended responses provided helpful suggestions for enhancing the model.

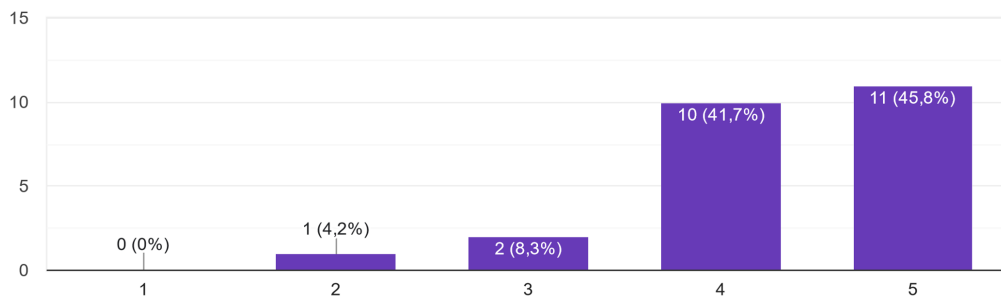


Figure 6.9: Does this model properly include all the necessary concepts and relations regarding Equality?

Several participants proposed additional concepts and relationships to be included, such as incorporating aspects of religion, connecting gender roles and norms more closely with culture, and expanding on race and sexual orientation as influenced by geographic location and belief. Others suggested adding a concept related to art with relationships to *Action* and *Sociocultural Context*. Additionally, one participant proposed the "Access Culture" concept to promote physical, social, and intellectual access to cultural participation. Another commented that while the model focused on personal aspects, it should consider environmental factors, ethics, and contextual values to provide a more comprehensive approach. All these answers should be considered in future work to improve and expand the culture part.

One participant said: "I didn't understand well what the *Action* meant in terms of Culture". From this comment, it's clear that the description of the *Action* concept needs to be clearer in the framework. Responding to this comment, it's essential to clarify that actions are influenced by cultural context, impacting how users approach tasks. Collectivist cultures, like China, may prefer teamwork-oriented software, while individualistic cultures, like the U.S., favor tools that promote personal autonomy. For example, Japan requires mobile phones to make a sound when taking photos to respect privacy norms. Thus, the concept of *Action* includes both functional and cultural elements.

The feedback indicates a need to refine and expand the Culture part of the model to capture better the diverse cultural elements. Taking into account the responses from the participants is a step towards that goal.

6.2.2.4 Gender

The evaluation of the Gender part of the model was mainly positive, as shown in Figure 6.11, with 12 participants (50%) rating it a 5 (Strongly Agree), 9 (37.5%) rating it a 4, 2 (8.3%) giving it a 3, and only 1 (4.2%) rating it a 1, indicating that most found the model satisfactory but pointed out options to refine it.

Only one answer wasn't constructive, so it was discarded as it couldn't be used to improve the model. Other participants suggested improvements to the model. One

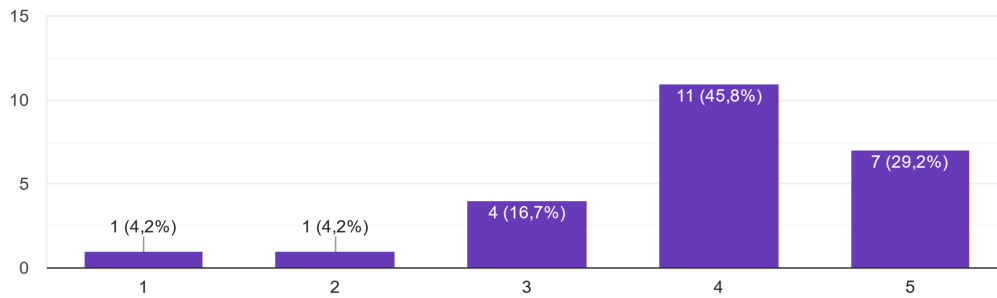


Figure 6.10: Does this model properly include all the necessary concepts and relations regarding Culture?

proposed: "I think people's perspective on their own gender can change the way they see other genders, this could be reflected in the diagram by adding a relationship from "Perceptions" to (for example) "Gender Stereotypes".", while another suggested the inclusion of "Professional Characteristics," such as soft skills.

Another comment suggested: "I would say that gender norms are the ones that create gender stereotypes rather than gender roles, the relation between biological sex and gender roles seems backward to me since biological sex is "set in stone" that is what defines our "roles".". The first part of the comment regarding gender norms creating stereotypes is acknowledged. However, it can be argued that gender roles are more involved in the creation of gender stereotypes. Gender roles define specific expected behaviors and characteristics associated with masculinity and femininity, which, in turn, reinforce stereotypes about how individuals should act based on their gender. The second part of the comment supports the model's view that gender roles are based on biological sex. In this context, biological sex shapes societal expectations and norms, illustrating how these roles are formed and maintained.

One participant noted the need to balance the model's parts. While it addresses sociocultural context, it doesn't connect to norms and laws. The participant points out that it's important to ensure that access remains a main focus throughout the model, which is a valid recommendation.

The feedback on the Gender part of the model emphasizes its strengths while suggesting improvements. Addressing these will improve the model's overall effectiveness.

6.2.2.5 Software System

The evaluation of the Software System part of the model produced mostly positive results, Figure 6.12. Out of the 24 participants, 9 (37.5%) rated it a 5 (Strongly Agree), 11 (45.8%) rated it a 4, 2 (8.3%) participants rated it a 3, and 2 (8.3%) others rated it a 2, with no one rating it a 1. This indicates that many find it satisfactory.

Several participants provided constructive feedback. One questioned: "What is the objective of Software Systems and its relationships in your proposal? I think it will be unnecessary as it is a transversal element, which is not represented in the relationships

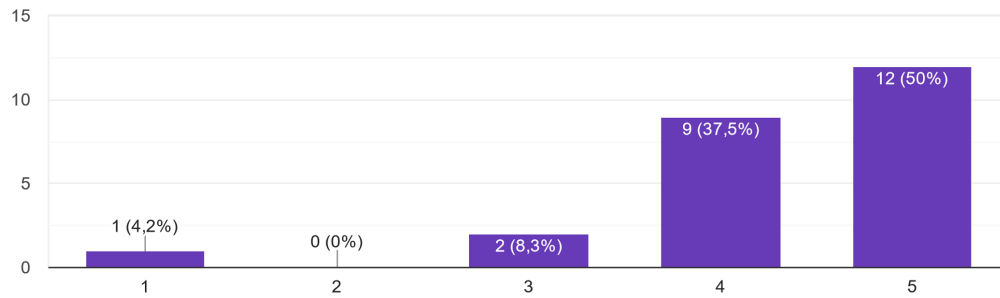


Figure 6.11: Does this model properly include all the necessary concepts and relations regarding Gender?

you propose, according to my definition". This suggests that the concept of the Software System does not need to be detailed as a separate entity. Another commented: "This part of the model seems little, but it can be fine depending on its purpose". So, further clarification is needed regarding the connections within the model to ensure that the role of Software Systems is well understood.

Another participant cautioned against the completeness fallacy which suggests that a model must include every detail to be effective. The participant said that its current version is a good starting point. The concern was that focusing on the UML diagram too much could distract from important human values related to quality. The participant suggested approaching from the perspective of Non-Functional Requirements (NFRs) or "soft goals" to offer better insights. This feedback is a valuable recommendation for future improvements to the model.

To enhance the model, it's essential to clarify the role of Software Systems and its connections.

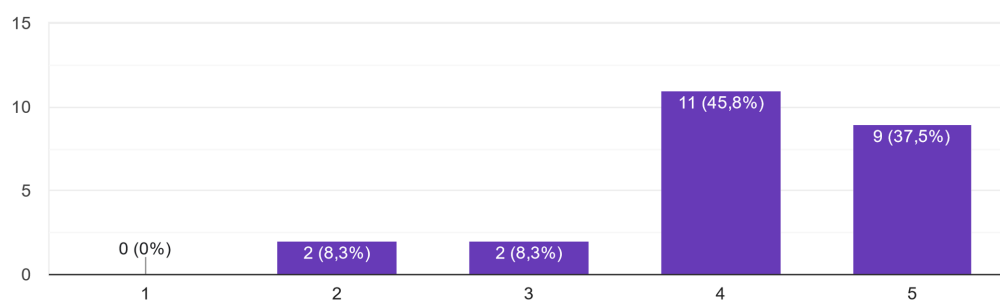


Figure 6.12: Does this model properly include all the necessary concepts and relations regarding Software System?

6.2.2.6 Human

The evaluation of the Human part of the model was generally positive. Figure 6.13 shows that 12 (50%) participants rated it a 5 (Strongly Agree), 9 (37.5%) rated it a 4, 1 (4.2%) gave it a 3, and 2 (8.3%) rated it a 2, with no one rating it a 1.

This section only has one answer, the participant noted that while the model focuses on users, it should also consider humans indirectly affected by the system, not just those interacting with it. They highlighted that this happens when a Human-Computer Interaction (HCI) perspective dominates, whereas a broader view, like in Requirements Engineering (RE), should also be considered. This feedback encourages expanding the model to consider a broader human impact apart from users directly affected by the system.

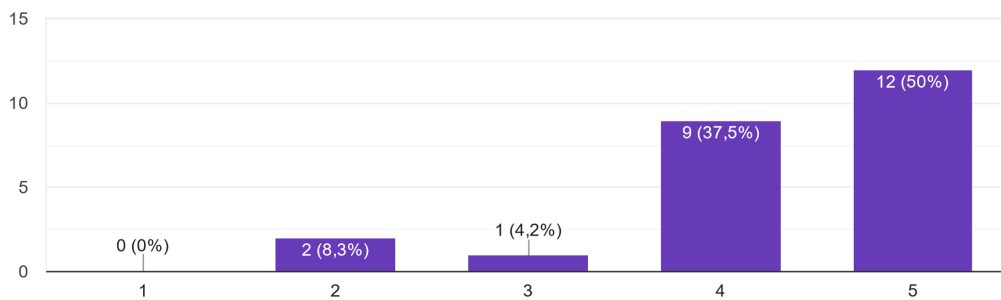


Figure 6.13: Does this model properly include all the necessary concepts and relations regarding Human?

6.2.3 Conceptual model

In this section, after evaluating the model's parts individually, the participants were asked about their overall opinion of the model. For this purpose, six questions were presented.

The first, which concerns the usefulness of the conceptual model, was a Likert scale question where 1 means the participant found the model not useful and 5 means that he found it very useful. The second question, which pertains to the difficulty in understanding the model, was similar to the first, where 1 means the participant found it not difficult and 5 represents very difficult. These were relevant questions to assess whether users would be comfortable applying the model.

The third used a scale from 1 meaning strongly disagree and 5 strongly agree to question if the participant would use the model to develop an application where universalism values would be relevant. The fourth question was whether the participants would recommend the model, using a scale from 1 meaning don't recommend and 5 meaning strongly recommend. The final two questions were open-ended and optional, asking about the strongest and weakest points in the model. These serve to further understand the participant's opinions.

6.2.3.1 Usefulness of the conceptual model

Figure 6.14 shows that there were mixed responses, however, most were concentrated on the useful and very useful categories, meaning that the model is useful as intended.

Out of the 24 participants, 10 (41.7%) rated the conceptual model as very useful, and 8 (33.3%) rated it as useful, indicating that 75% provided positive feedback. 3 participants (12.5%) considered it neutral, while 2 (8.3%) rated it as somewhat useful, and only 1 participant (4.2%) rated it as not useful.

Interestingly, when comparing these results with the previous question regarding participants' perception of the importance of universalism values, the data reveals that 11 participants kept their original ratings, 6 reduced their score by one point, 6 increased their rating by one point, and 1 participant increased their rating by two points. This suggests there was some shift in participants' perception of the model's usefulness as they progressed through the evaluation. Regardless of the slight shift, the overall results remained consistent and positive.

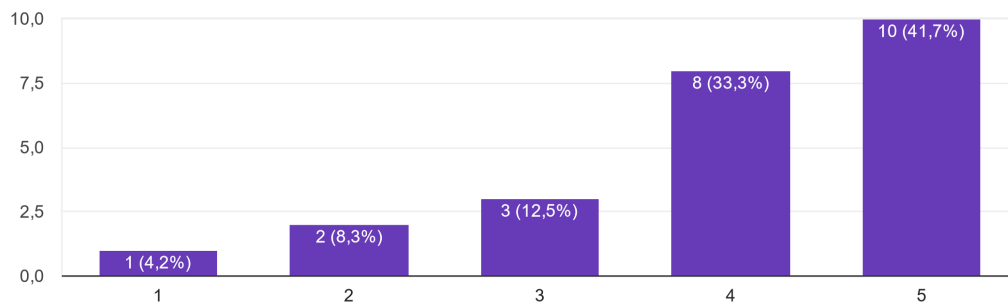


Figure 6.14: How useful is this conceptual model?

6.2.3.2 Difficulty in understanding the conceptual model

Figure 6.15 shows that there were mixed responses. The results for the difficulty of understanding the conceptual model indicate that participants generally found it slightly to moderately difficult to understand. The graphic shows that 3 (12.5%) participants responded that the model was not difficult to understand, with 7 (29.2%) considering it slightly difficult to understand. The largest group, 8 (33.3%) participants, found it neutral. Additionally, 5 (20.8%) participants rated the model as difficult, and 1 (4.2%) as very difficult.

An important observation is that all participants who rated the model as difficult or very difficult, with scores of 4 and 5, had 0-2 years of experience in conceptual modeling, indicating that less experienced participants had more trouble comprehending the model. This could be because of the complexity of the concepts and their relationships, which made the model have a large scope, and in consequence, made the model more difficult to read.

In summary, while most participants found the model manageable, the feedback indicates that users with less experience may require extra support or simplifications to completely comprehend the model. Future adjustments could be made to improve clarity for less experienced users.

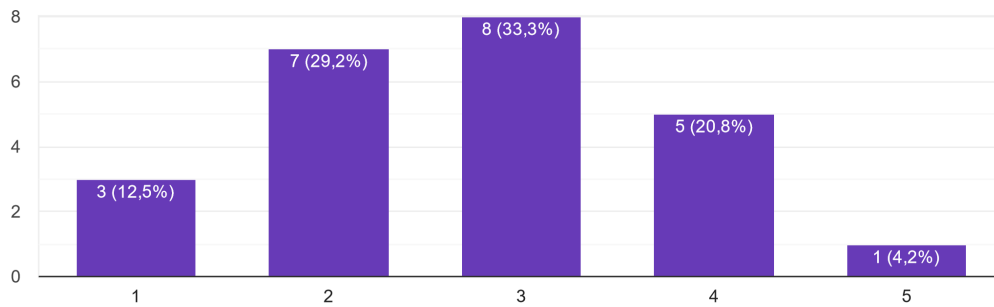


Figure 6.15: How difficult is it to understand this conceptual model?

6.2.3.3 Using the conceptual model to develop an application

Figure 6.16 shows that the results were mostly positive about whether the participants would use the model. Results reveal that 5 (20.8%) participants answered strongly agree, 13 (54.2%) said agree, 5 (20.8%) selected neutral, 0 participants responded with disagree, and only 1 (4.2%) said strongly disagree.

Interestingly, the participant who rated it a 1 also rated the importance of universalism values and the usefulness of the model as 1 in previous questions. This participant has a PhD but only 0-2 years of experience with conceptual modeling and rated the difficulty of the model as 4, difficult. This data indicates that this participant most likely struggled to understand the model due to limited experience in conceptual modeling. The low rating on the importance of universalism values may reflect the participant's priorities that differ from the model's focus, which influenced their general evaluation.

The data indicates that despite the previous mixed responses regarding the difficulty of understanding and the usefulness of the model, most participants would still consider using it to develop applications where universalism requirements are relevant. This means that regardless of challenges, the model is generally seen as a useful tool.

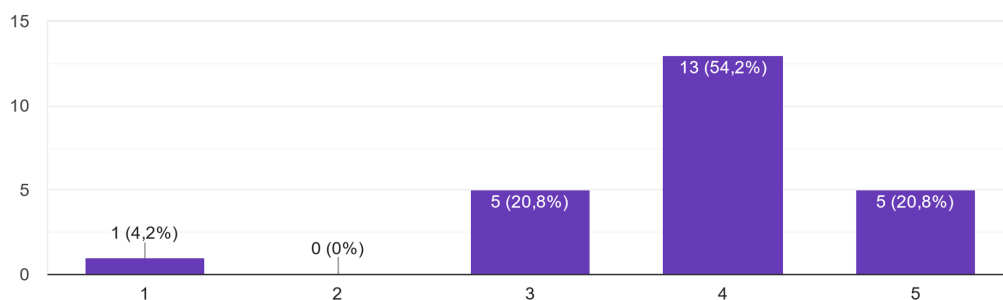


Figure 6.16: Would you use this conceptual model to develop an application where universalism requirements are relevant?

6.2.3.4 Recommending the conceptual model

Figure 6.17 shows the results, mostly positive, about the participant's willingness to recommend the model. The results reveal that 6 (25%) participants said they strongly recommended it, 10 (41.7%) said they recommend it, and 7 (29.2%) were neutral, with no participants selecting the not recommended option, and 1 (4.2%) said they strongly didn't recommend.

In this case, the participant who rated the model a 1 in the previous question also gave the same rating here. This confirms the participant's consistency in their evaluation of the model.

Nevertheless, most participants still indicated that they would recommend the model, which highlights its value.

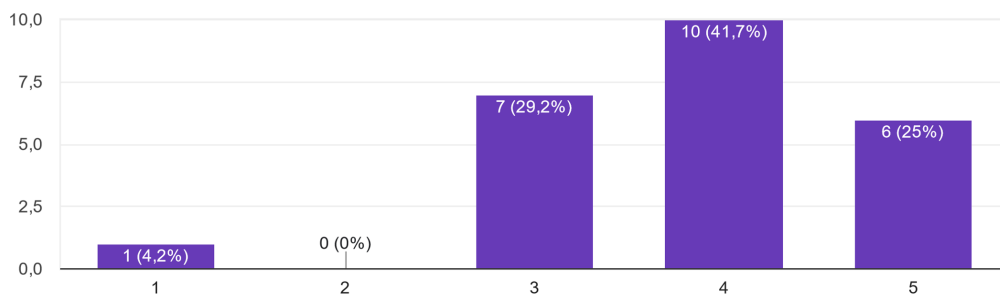


Figure 6.17: Would you recommend this conceptual model?

6.2.3.5 Strongest points in the conceptual model

A total of 19 (79.2%) participants shared their opinions when asked about the strongest points of the model, indicating an elevated level of interest in further exploring its potential. The feedback was overwhelmingly positive and varied, highlighting the model's effectiveness in visualizing complex concepts related to universalism.

The participants praised the model for its effectiveness in visualizing complex universalism concepts and making the connections easy to understand. The structured presentation, including color coding and division of parts of the model, was said to have facilitated the comprehensiveness. The feedback received is listed below:

- "It's easy to draw the connection between the various concepts of universalism that without a visualization like this would be difficult to connect. The categories make sense and are clearly defined."
- "Approach to Gender, such a sensitive concept in today's society."
- "A lot of good requirements are specified and well related, I also think the model's organization deserves praise, the placement and color coding of different requirements makes it easy to quickly understand."

- "I believe it lists the basic concepts of Equity and Diversity and builds relations between them."
- "For the sections that it represents it seems to define clearly and in a good manner how each part is connected to the next and which parts are relevant".
- "Very detailed and considers a lot of relevant aspects".
- "Not Too Complex, well structured and designed (ex: different colors for different categories)".
- "A more complete and sensible to all dimensions considered system will be developed."
- "It is easy to understand the connections between each concept".
- "In my opinion, there are two strong points: the concepts are clearly described, and the usefulness of the relations between the concepts."
- "Components well distinguished".
- "Great to map out requirements (social, values...) that otherwise could easily be forgotten during development".
- "The completeness and high organization of this model into different concepts that can be viewed as a whole or a subset (in case only some universalism concepts are required for a specific application)".
- "Its comprehensiveness".
- "Easy to read and understand".
- "The division of multiple categories and what it's included in each makes it easier to understand the concepts relations".
- "A escala a que tenta atuar/bastante amplo", meaning: the scale on which it tries to operate/quite broad.
- "Unified view of universalism requirements".

One participant even provided feedback to improve the model: "The possibility of proposing relations among concepts. You could have used the package diagram to better modularize the whole diagram."

Overall, the feedback indicates that the model is accessible and useful, providing a unified view of universalism concepts. This feedback is encouraging and provides helpful information on what diverse participants thought of the model.

6.2.3.6 Weakest points in the conceptual model

Regarding the model's weakest points, A total of 16 (66.7%) participants shared their opinions, and the feedback from participants can be grouped into two main categories. The first focuses on the model's complexity, with several participants reporting that the quantity of concepts makes it feel cluttered and overwhelming. This complexity can lead to difficulties in navigating the relationships between components, especially for individuals with beginner to intermediate levels of experience in modeling. This feedback is listed below:

- "It feels a bit cluttered but it's only natural considering how many concepts are present in it."
- "Due to its detail, it can be difficult and time consuming to understand the different relationships".
- "Arduous for individuals, who have beginner to intermediate levels of experience in modelling, to read, comprehend and apply its power to the production of their artefacts."
- "The learning curve, since there are a lot of definitions and some relations between the concepts are difficult to understand, and therefore to use/apply."
- "Complexity".
- "Big conceptual models can be hard to navigate and understand relationships between components".
- "Not the model itself, but the willingness of professionals to accept (and understand) the importance and impact of every identified concept."
- "Torna se bastante complicado tentar acompanhar cada linha ou caminho", meaning that it becomes quite complicated trying to follow each line or path.

The second category relates to specific areas that could benefit from further investigation in future work. This feedback is listed below:

- "In the future, an Environment section may be included."
- "I don't completely agree with most of the "human centered" requirements being directly related to "Human", which only "interacts" with the Software System. I think especially nowadays the software system interacts with and changes the Human, therefore we need to build systems that take this into account. Inherently aspects such as "Values" and "Norms" need to be related to the software system as much as they need to be related to the human, even if these values or norms are to maintain neutrality (and by extension consider no values and norms)."

- "It only links the concepts to software dev at a very shallow level."
- "Some of the parts of the model between different sections should be more connected with each other rather than being isolated from each other".
- "The lack of focus on the qualities as goals, which would give more possibilities to map and correlated different qualities."
- "Possible model incompleteness (it depends on its scope and purpose)".

Other participants also said:

- "I didn't understand well what the Action concept meant when in connection with the concept Culture".
- "Although it does frame the issues it intends to address quite well, I don't think it will be that useful in creating solutions".

Despite these comments, the model was designed to provide a representation of universalism values in software development. It seeks to encourage requirements engineers to adopt a value-based perspective during elicitation. It would be beneficial to conceptualize the model to improve its comprehensiveness and facilitate collaboration with non-technical stakeholders. Overall, the responses indicate genuine interest in the topic and potential for further development.

6.2.4 Prototype Evaluation

After evaluating the model, comes the prototype tool evaluation. The participants were asked about the different pages of the tool and their respective functionalities. Then, they had to give their general opinion of the tool. Finally, the questionnaire concludes with questions about the framework document, the guide, and the video. For this purpose, six questions were presented.

6.2.4.1 Tool page

The tool page received positive responses about the ease in exploring the model and the selection and saving of concepts.

In terms of exploring the model, Figure 6.18 shows that, out of the 24 participants, 12 (50%) rated it as very easy, 10 (41.7%) rated it as easy, and only 2 (8.3%) rated it as neutral. No participants reported significant difficulties, indicating that most thought that exploring the model was straightforward.

Despite the positive results, three participants provided constructive feedback on difficulties they had and future improvements. One participant pointed out two modifications: "Initially, when I clicked on a model section, due to the lack of feedback, I was not aware that the page had changed since the changes appear at the bottom of the webpage, which

is not visible before scrolling down. It could be useful if after a user clicks on a section, the web app automatically scrolls down to reveal the characteristics, or have some sort of visual or auditorial feedback. An unnecessary and superficial change, but the "What is this tool?" and "See instructions" buttons could have a "Go back" button at the bottom of the information, instead of having to use the slightly "hidden" "Home" button." The first part of the comment about how the participant did not realize that the characteristics table appeared at the bottom of the page is a relevant point and automatically scrolling down is a possible solution for this problem. Implementing this feature would also solve another participant's comment: "A bit annoying to have to always scroll up and down to see the definition of what i clicked on". The second part of the first participant's comment suggests adding a go-back button on the pages about the tool and instructions. Lastly, one participant proposed implementing a direct search feature for concepts to facilitate navigation.

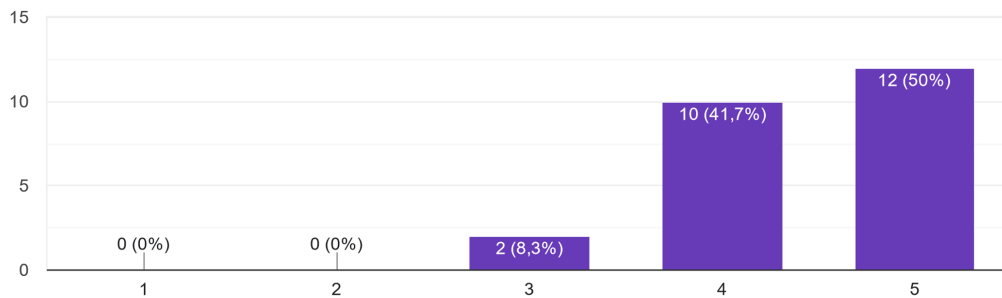


Figure 6.18: About the tool page, did you have any difficulty in exploring the model?

In terms of selecting and saving the concepts, Figure 6.19 shows that, of the 24 participants, 11 (45.8%) rated their experience as very easy, while 10 (41.7%) found it easy. Only 3 (12.5%) participants rated it as neutral. No participants reported significant difficulties, reflecting a general ease of use.

Five participants provided specific feedback on challenges they faced. Two participants commented that: "Documents are only really "created" after the questions related to each requirement are answered, I think they could be saved beforehand so that the user can add all the desired requirements, load their document, and continue adding more/answer the questions then.", and "For the reasons mentioned above, I kept going to the "Documents" tab to create a new document first (which I thought had to be the first step), and clicked model sections. Since I did not notice changes, I retried the process a few times until I decided to rewatch the video.". These comments suggest letting the users save the document first and then add the concepts and questions. This could be an additional functionality, this way the user can choose to save a document before or after the process.

One participant mentioned that the save button and characteristics were not immediately visible, requiring them to scroll down, which caused some confusion. This comment was supported by another participant who said: "It is easy to save a concept but I had some difficulty to get to the button 'Add To Document'. I think it would be easier to

show the button near the top of the page and move the 'Next' button to the top-right corner of the page.". Moving the next button could lessen the confusion, however, adding the automatically scrolling down option could solve the problem of accessing the save a concept button. While the suggestion of moving this button is acknowledged, having the button near the characteristics could be more beneficial.

Lastly, one user reported a minor bug with the characteristics table below the selection area but indicated that these issues were not significant. In the future, this would need to be rectified.

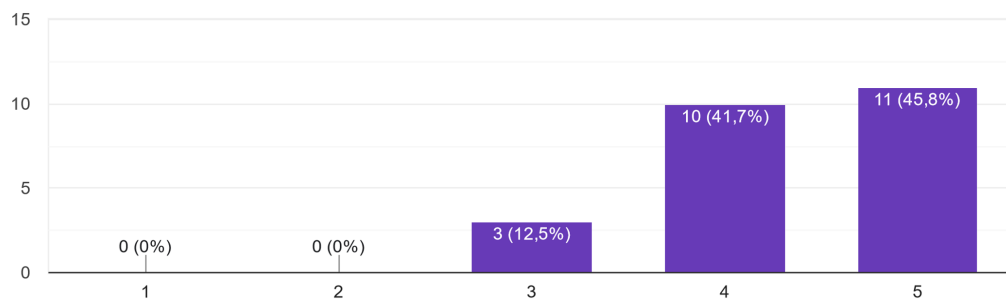


Figure 6.19: About the tool page, did you have any difficulty in the selection and saving of the concepts?

All in all, the tool page was well-received and these suggestions could further improve its usability and user experience, however, these improvements would need to be validated with another evaluation.

6.2.4.2 Questions page

The questions page received predominantly positive responses related to selecting and removing concepts and the case example.

In terms of selecting and removing concepts, in Figure 6.20, 14 (58.3%) participants rated their experience as very easy, and 8 (33.3%) found it easy, with only 2 (8.3%) participants rating it as neutral. No participants reported significant difficulties.

Only two participants provided comments. One user noted: "It's a bit difficult to add other concepts after going into the questions page going back and forth between the two pages, having a list on the same page as the map that allowed the user to remove the concepts they didn't want anymore without swapping pages would help a lot in that regard". Adding a feature to provide a list alongside the map would improve usability, thus this is a good suggestion.

After examining the spreadsheet, it was revealed that the one participant who previously rated as neutral their difficulty in selecting and removing the concepts mentioned they did not explore the removal functionality. This means that there might be areas of the tool that some participants did not fully engage with.

In terms of the usefulness of the case example of the app ShareTheMeal, in Figure 6.21, most participants found it beneficial, with 10 (41.7%) participants rating it as very useful,

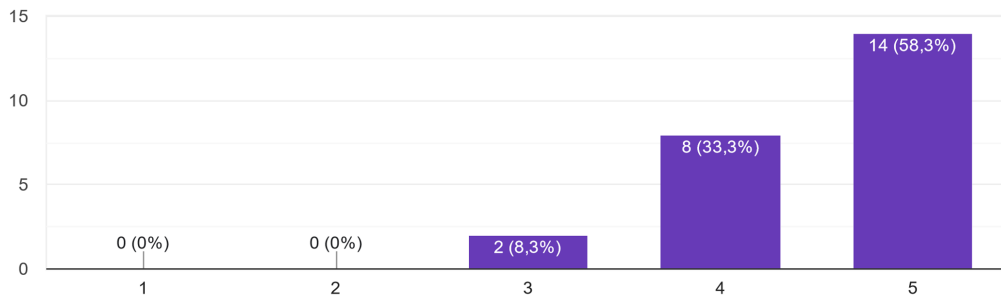


Figure 6.20: About the questions page, did you have any difficulty in selecting and removing the concepts?

11 (45.8%) rating it as useful, and only 3 (12.5%) giving a neutral score. No participants rated the example as unhelpful.

However, there is room for improvement, as four participants provided additional feedback. One participant pointed out redundancy in the case example of ShareTheMeal, noting that some of the statements seemed to be repetitive and didn't provide additional clarity. Two other participants said they would prefer more practical demonstrations in the case example. While it was useful, they felt it lacked concrete examples of how the tool's concepts are applied in real-world scenarios. These comments indicated that the case example needs improvement to provide a clear illustration of the requirements examples.

One participant was confused over a link that redirected them to the ShareTheMeal website instead of a model-related example. This implies that further clarification is required to make it clear that the link is intended to introduce users to the app and its purpose, helping them better understand the context behind the requirements examples.

Lastly, one participant gave an additional suggestion: "On a side note, an unimportant detail, but I think that "Case Example: ShareTheMeal" and "Return to Questions" should be buttons instead of links, since they do not lead to separate webpage, to be consistent with the rest of the interface".

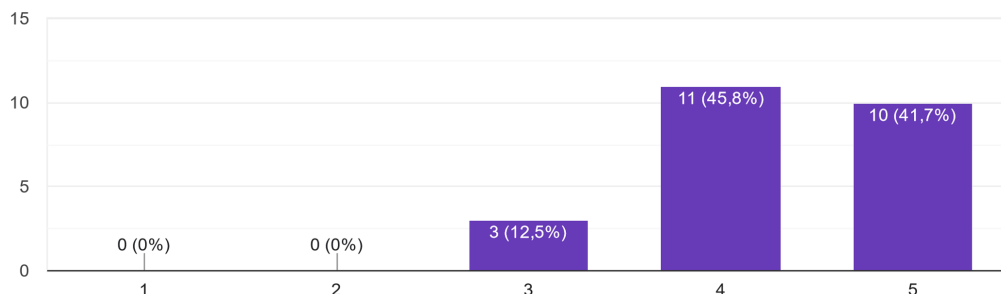


Figure 6.21: Was the case example of the app ShareTheMeal useful?

In general, most participants found the questions page to be user-friendly but improving the case example and the design could improve the experience.

6.2.4.3 Document Page

The document page received predominantly overwhelming feedback, related to adding and removing participants. Figure 6.22 shows that 18 (75%) participants rated it as very easy, 4 (16.7%) said it was easy, 1 (4.2%) selected neutral, and only 1 (4.2%) said it was very difficult.

One participant did mention an issue where, despite creating documents, they did not appear as a document. This indicates that there might be a bug or confusion with the interface, which might require clarification in the guides provided or troubleshooting. Despite this one comment, the reception of this page was highly promising.

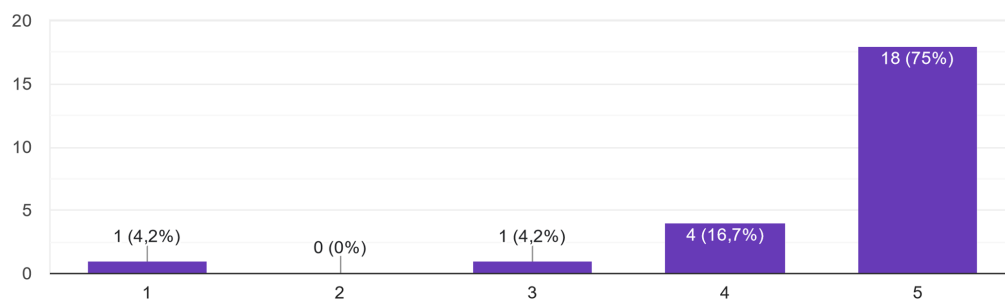


Figure 6.22: About the document page, did you have any difficulty in adding and removing participants?

6.2.4.4 Documents Page

The following documents page also received highly positive feedback, related to exporting, editing, or deleting a document. Figure 6.23 shows that 17 (70.8%) participants rated the process as very easy, 5 (20.8%) said it was easy, 1 (4.2%) selected neutral, and only 1 (4.2%) said it was very difficult.

The one participant who rated 1 also rated the previous question as 1, reporting that they couldn't use the feature as the created documents did not appear. Another participant said that the edit button should redirect to the beginning of the process rather than the end, which should be considered to improve the navigation.

6.2.4.5 General opinion about the tool

The general opinion about the tool is mostly positive feedback with nine participants suggesting how to improve the tool.

In terms of how intuitive the interface of the tool is, Figure 6.24 shows that 7 (29.2%) participants rated it as very intuitive, 14 (58.3%) as intuitive, 2 (8.3%) as neutral, and 1 (4.2%) participant found it confusing.

In terms of the navigation of the tool, Figure 6.25 shows that 7 (29.2%) participants rated it as very easy, 14 (58.3%) as easy, and 3 (12.5%) gave it a neutral rating, with no one finding it difficult.

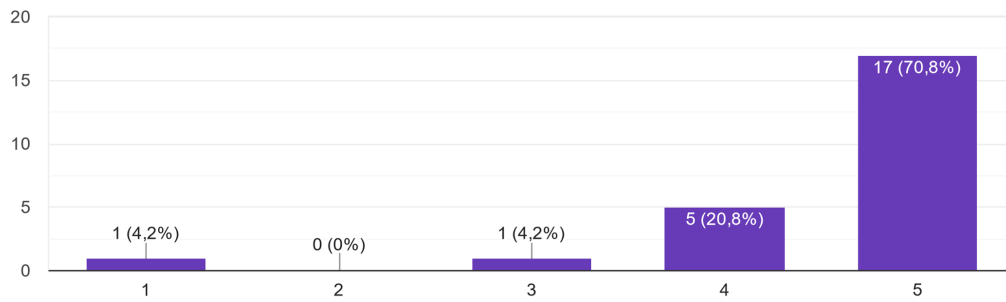


Figure 6.23: About the documents page, did you have any difficulty in exporting, editing or deleting a document?

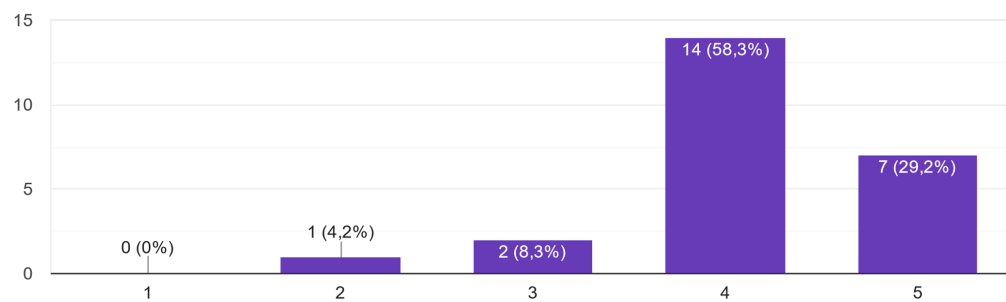


Figure 6.24: How intuitive would you rate the interface?

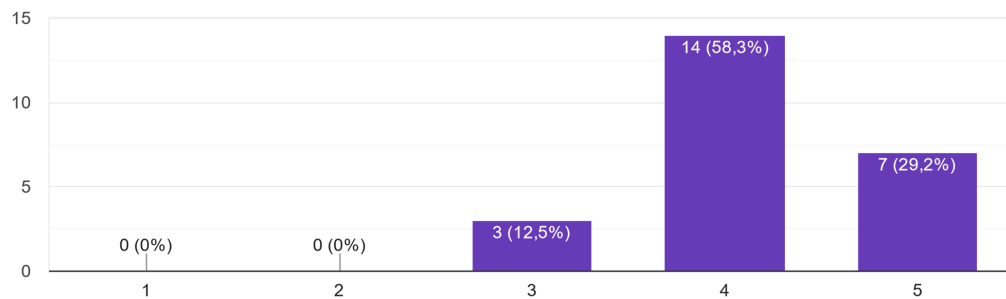


Figure 6.25: How is the navigation of the tool?

With the open-ended question, nine participants offered constructed suggestions for improvements. These are presented below:

- "I gave this suggestion earlier but having the list of concepts on the same page as the map would help as it wouldn't force the user to go back and forth between pages just to handle concept addition and removal".
- "The button "Next" being placed close to the selected requirement might make the user think that by clicking it they are selecting the next requirement.".
- "The box with the models is too large in my opinion, i think if the box was smaller you could show the characteristics on the side of the box for easier reading while navigating".

- "Seems to be an interesting and useful project, even if the usefulness may be circumstantial, as Universalism may not be a priority for some projects, as users may focus more often on the Security and Privacy section. Not sure if it is useful or feasible, but attributing grades/scores to the documents based on how many questions are checked regarding a customizable user preferences and priorities. Could be useful to add a section regarding "UN's sustainable development goals", as a means to check if the project/document contributes to any of the goals, possibly presented in the form of questions like the rest of the concepts".
- "Congratulations! I really liked the idea! Please verify if the tool complies with the Web Content Accessibility Guidelines (WCAG). I also suggest organizing the diagram elements more clearly and specifying what the relationships between them represent (e.g., refinements, dependencies). Additionally, I'm unsure why certain elements are repeated. Since 'Software Systems' and 'Gender Inclusion' are represented in the same way (as rectangles), does that imply they are considered equivalent from a requirements perspective?".
- "Maybe a way to directly look up a concept could help navigation".
- "Include technical requirements and standards, rather than broad questions only, where applicable: Is this web page accessible? How is this measured? Does it fulfill the WCAG 2 guidelines? Providing technical guidelines and evaluation tools will help to assess these concepts more accurately."
- "I have already given them. But overall the tool is very good".
- "The part of the questions is a plus. However, from my experience, it was not clear if the questions are generated or are generated and proposed as you input a new concept or are provided by the user who added the concept."

Altogether, the tool received positive feedback, with suggestions that point to clear opportunities for refining and improving its functionality. The results display the participants' interest in this tool and the prototype's success in intuitiveness to create a universalism requirements document during the elicitation phase of a project.

6.2.4.6 Framework Document

The feedback regarding the framework document is predominantly positive. Participants found the document valuable for understanding the model and considered the description of the concepts largely comprehensible.

In terms of the document's usefulness for understanding the model, Figure 6.26 presents 10 (41.7%) participants indicated it was very useful, 10 (41.7%) also found it to be useful, and 4 (16.7%) rated it as neutral. No participants rated it as not useful.

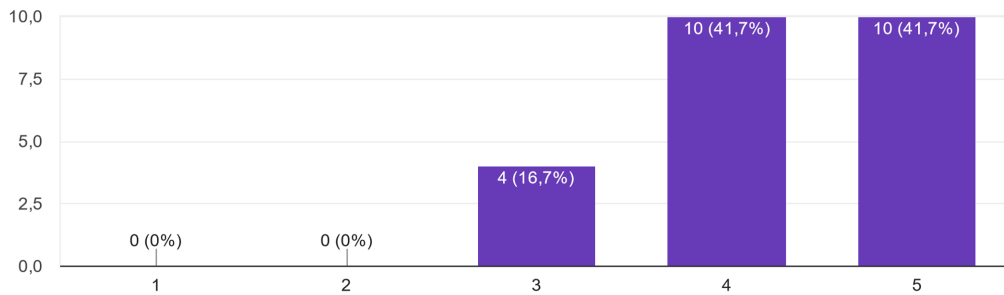


Figure 6.26: Was the document useful for your understanding of the model?

In terms of the comprehensibility of the concepts, Figure 6.27 shows 10 (41.7%) participants rated it as very comprehensive, 12 (50%) considered it comprehensive, and 2 (8.3%) rated it as neutral. There were no ratings of not comprehensive.

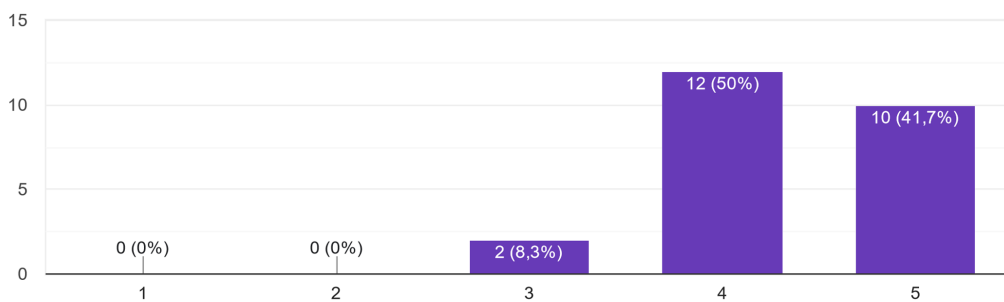


Figure 6.27: In general, was the description of the concepts comprehensible?

In general, these results demonstrate that the framework document effectively supports participants in understanding the model and clearly conveys the concepts.

6.2.4.7 Guide Evaluation

Both questions for this evaluation were optional, as the participant could have chosen to read the document or watch the video. The feedback regarding the document guide is predominantly positive as participants found the guide valuable and easy to understand.

In terms of the guide's usefulness, Figure 6.28 presents that 8 (47.1%) participants indicated it was very useful, 8 (47.1%) also found it to be useful, and 1 (5.9%) rated it as neutral. No participants rated it as not useful.

Regarding the comprehensibility of the guide, Figure 6.29 shows that 8 (47.1%) participants rated it as very easy to understand, 7 (41.2%) considered it easy to understand, and 2 (11.8%) rated it as neutral. There were no ratings of not easy to understand.

In general, these results demonstrate that the document guide effectively supports participants in performing their tasks on the tool.

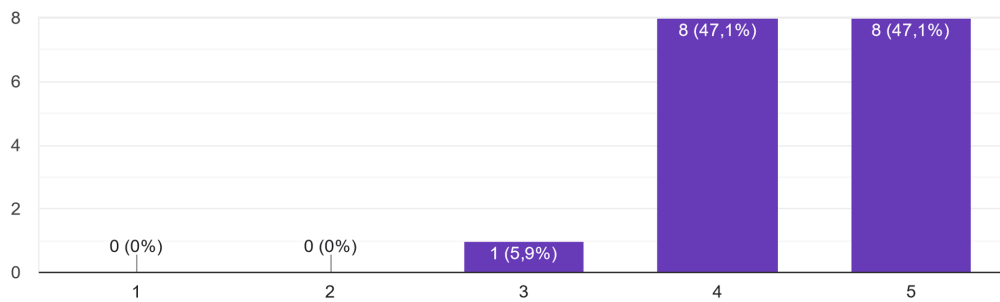


Figure 6.28: Was the guide useful?

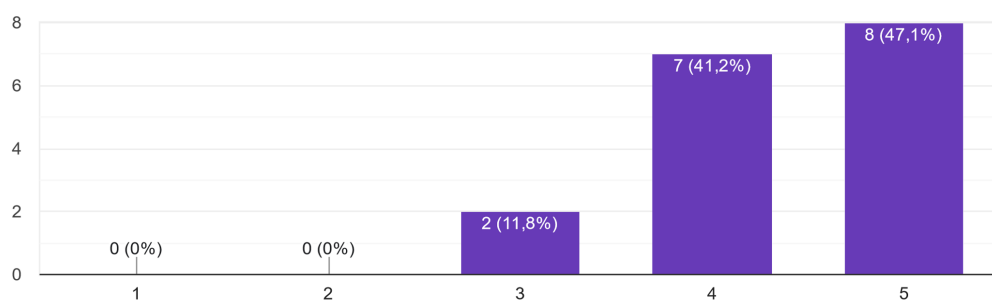


Figure 6.29: Was the guide easy to understand?

6.2.4.8 Video Evaluation

The question for this evaluation was optional, and the feedback regarding the video was predominantly positive. Participants found the video valuable in supporting their understanding.

Figure 6.30 shows that 11 (64.7%) participants indicated it was very useful, 4 (23.5%) found it to be useful, and 2 (11.8%) rated it as neutral. No participants rated it as not useful.

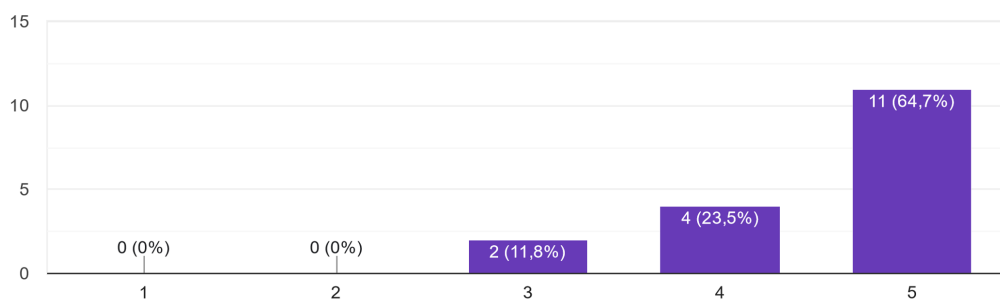


Figure 6.30: Was the video useful?

The guide and video evaluation sections had 17 answers and analyzing the spreadsheet revealed that 13 used both resources, 4 only used the document guide, and 4 only watched the video. Out of the 24 participants, only 3 did not use these resources.

Overall, these results demonstrate that the video effectively assists participants and is regarded as a valuable resource.

6.3 Threats to validity

Following the evaluation of the conceptual model, this section discusses the threats to its validity. The proposed guidelines in [106] were followed to identify them. They are divided into internal, external, construct, and conclusion validity.

6.3.1 Internal Validity

An issue that may occur in participants of the evaluation is that they may feel overwhelmed if they don't know enough about the topic in question, in this case, universalism values and conceptual models. To mitigate this problem the survey was designed to be as concise and coherent as possible without removing all the essential areas that needed to be covered, such as the various parts of the model, the prototype tool, and the guides. To assess the participant's knowledge of the subject, in the pre-survey, questions were asked regarding the participants' scientific area of degree, the years of experience with conceptual modeling, and the importance they assign to universalism values during software development. The survey was even separated into sections and split the model into different parts to evaluate them one at a time, to ease the participant. However, the number of sections might affect the quality of their responses due to fatigue. To counter this, the questions of the sections related to the model were kept short, with two questions per model part. The section with the most questions was the prototype tool section, with multiple Likert scales and open-ended questions. To appease the participants, all open-ended questions were optional. These efforts seem to have paid off and the threat may have been mitigated since all open-ended questions had various answers, some more than others.

6.3.2 External Validity

Many of the participants claimed they had little experience with conceptual modeling (0-2 years), as shown in Figure 6.6. However, that doesn't mean that their answers were less appreciated when in comparison to the rest of the participants. The participants belonging to the 0-2 years category can be considered potential stakeholders in the requirements elicitation process since most do not have much expertise in this area, thus their responses are essential. The model is supposed to be comprehensive even for those with less experience. However, these participants may not have the necessary knowledge to give a thorough analysis of the model. Additionally, those with more experience in conceptual modeling may not be aware of universalism values and their concepts. Consequently, the number and quality, related to the knowledge in these areas, of participants should be revised to produce a more reliable collection of results. To

mitigate this threat, the prototype tool section of the evaluation provided a video and document guide to familiarize the participants with the tool, these guides explained how to use the tool and its functionalities. Most of the participants considered both guides useful and easy to understand, so this threat did not compromise the results of the tool. Regardless, the question concerning the importance of universalism and its sub-values served to assess if the usefulness and difficulty in understanding the model were because the participant was not acquainted with these concepts or because they thought the model itself was not useful or difficult to understand.

Another important point that needs mentioning is that all the material, e.g., form and guides, were in English to reach more people, yet, most participants did not have English as a native language so that could affect their interpretation of the concepts, model, and tool. Nonetheless, the participants gave comprehensive English questions to open-ended questions and considered the guides useful, only one answered in Portuguese. So, the decision to use English did not seem to have negative impacts.

6.3.3 Construct Validity

One potential problem that is always a possible threat to the validity of every survey is whether the structure of the questions was constructed perfectly to transmit what was intended, or even if they were the most appropriate ones to assess the study's objective. Nonetheless, the questions were carefully assembled, prioritizing clear phrasing, simplicity, and neutrality to reduce ambiguity. The form was also divided into distinct, organized sections to provide the participants with the necessary context for them to respond.

6.3.4 Conclusion Validity

In this evaluation, because the size of participants that answered the survey was not large, it can raise concerns regarding the validity of the evaluation results. Still, the number of responses was considerable, considering the time provided to participate was not much. It's risky to draw conclusions when the sample size is limited. To correct this and get further perspectives to assess the validity of the results obtained with this survey, another evaluation with many more participants, especially experts in human values and conceptual modeling.

6.4 Summary

This chapter presented the evaluation of the conceptual model of universalism requirements and the prototype tool created to assist the elicitation process. First, we introduced the evaluation planning, explaining its goal and the resources used to conduct the survey. The goal was to gather the viewpoints of others on the topic of universalism values in software engineering. For this, we created a questionnaire to collect participants' answers.

Then, we presented and discussed the results obtained from the 24 responses. There were positive results, some critiques, and suggestions. Most were constructive, being considerations for future research.

CONCLUSIONS AND FUTURE WORK

In this last chapter, we present the conclusions of the work carried out in this dissertation. We review our initial objectives and highlight what we consider to be the key contributions. Finally, we offer recommendations that may guide future research in this area.

7.1 Contributions

Equality and social justice are topics that more than ever need to be considered in software systems. If certain groups are deprived in favor of others due to a system not being designed correctly to reflect inclusivity and equitable values, then that raises unrest, user dissatisfaction, and negative consequences concerning socioeconomics. Systems must be fair for all users, regardless of their characteristics. Considering the importance of software in the present world, finding ways to address and handle equality problems is vital to ensure systems are aligned with and respect user's values. In this dissertation, we studied universalism problems in requirements engineering and proposed a Conceptual Model for Universalism Requirements.

The goal of the model is to support requirements engineers in eliciting universalism requirements by reducing the gap between human values and software development. The conceptual model was formed based on a systematic mapping study on universalism values in requirements engineering. The fundamental concepts were gathered from this process and served as the foundation for the proposal of the conceptual model.

To assist requirements engineers in eliciting universalism requirements, we introduced a framework that incorporates the conceptual model into their existing practices. To support this framework, we developed a prototype tool to facilitate the elicitation planning phase. Then, we conducted an empirical evaluation to assess the conceptual model's utility and comprehensibility. The encouraging feedback and results from this evaluation have set the stage for future research opportunities. Additionally, the evaluation gathered participants' views on the significance of universalism values in software development, further emphasizing the importance of this work.

We highlight the achievements below:

- A conceptual model for universalism requirements.
- A prototype tool to support the framework's elicitation planning phase.
- An empirical evaluation of the conceptual model and tool with 24 participants.

7.2 Future work

In this dissertation, we studied universalism problems in Requirements Engineering and constructed a Conceptual Model for Universalism Requirements. This model serves as an initial contribution to addressing universalism requirements, however much more needs to be done to integrate these values completely.

For future work, we outline potential approaches and opportunities to explore the conceptual model further:

- While we developed a prototype tool to support the application of our model, a more thorough evaluation would improve its effectiveness. The feedback from the evaluation brought out valuable insights to enhance the tool, suggesting areas that need improvement. After implementing the modifications based on the participants' suggestions, another evaluation will need to be conducted to validate the improvements.
- The framework proposed requires further refinement to strengthen its reliability since it is only an initial step to integrating the conceptual model. Also, we did not provide enough detailed documentation about the concepts of the model, which must be considered. Ideally, the framework should include methods to represent and examine universalism requirements properly.
- Concerning the evaluation of the conceptual model, although the results were overall positive, there were still comments highlighting weaknesses in the model that should be addressed, revealing that its concepts and relationships would benefit from further refinement. Every part could be improved, as the evaluation offered suggestions for all parts of the model. However, it would be most useful to start with the Culture part, which could be expanded to include a wider range of cultural factors and concepts, ensuring a more comprehensive representation of its development and usage in software. Also, the Equality part requires further refinement, particularly to clarify the relationships between fairness, education, and other elements to more clearly capture its meaning. To address further the Universalism value, an environment section should also be added. These would be the first steps to improve the conceptual model.
- Once the model has been refined, it will need to be reapplied to capture universalism requirements for more case studies. In addition, specific guidelines could be devised

to have a more detailed application of the framework, thus further validating the usefulness of the conceptual model and tool. Future studies could also explore how the model performs in software projects using agile methodologies.

BIBLIOGRAPHY

- [1] Y. P. C. Aguiar et al. "AutismGuide: A usability guidelines to design software solutions for users with autism spectrum disorder". In: *Behaviour & Information Technology* 41.6 (2022), pp. 1132–1150 (cit. on p. 17).
- [2] M. Al Hinai and R. Chitchyan. "Building social sustainability into software: Case of equality". In: *2015 IEEE Fifth International Workshop on Requirements Patterns (RePa)*. IEEE. 2015, pp. 32–38 (cit. on pp. 27–29, 35–39, 41).
- [3] M. Al Hinai and R. Chitchyan. "Evaluating Equality Requirements for Software Systems." In: *ICT4S*. 2018, pp. 20–36 (cit. on p. 35).
- [4] O. I. Al Mrayat, N. Norwawi, and N. Basir. "Requirements elicitation techniques: comparative study". In: *International Journal of Recent Development in Engineering and Technology* 1.3 (2013), pp. 1–10 (cit. on p. 7).
- [5] K. Bach and R. Harnish. "Linguistic communication and speech acts". In: *SER-BIULA (sistema Librum 2.0)* (1979) (cit. on p. 46).
- [6] S. Bardzell. "Feminist HCI: taking stock and outlining an agenda for design". In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. CHI '10. Atlanta, Georgia, USA: Association for Computing Machinery, 2010, pp. 1301–1310. ISBN: 9781605589299. DOI: [10.1145/1753326.1753521](https://doi.org/10.1145/1753326.1753521). URL: <https://doi.org/10.1145/1753326.1753521> (cit. on p. 51).
- [7] D. F. Beer. "Web Accessibility for People with Disabilities: An Introduction for Web Developers". In: (2003) (cit. on p. 17).
- [8] E. A. Bo Malmberg and J. Östh. "Segregation and Urban Unrest in Sweden". In: *Urban Geography* 34.7 (2013), pp. 1031–1046. DOI: [10.1080/02723638.2013.799370](https://doi.org/10.1080/02723638.2013.799370). eprint: <https://doi.org/10.1080/02723638.2013.799370>. URL: <https://doi.org/10.1080/02723638.2013.799370> (cit. on p. 35).
- [9] E. Britannica. *Gender Role*. 2023. URL: <https://www.britannica.com/topic/gender-role> (visited on 2024-07-27) (cit. on p. 48).

-
- [10] S. Brooke. ““Condescending, Rude, Assholes”: Framing gender and hostility on Stack Overflow”. In: *Proceedings of the Third Workshop on Abusive Language Online*. Ed. by S. T. Roberts et al. Florence, Italy: Association for Computational Linguistics, 2019-08, pp. 172–180. DOI: [10.18653/v1/W19-3519](https://doi.org/10.18653/v1/W19-3519). URL: <https://aclanthology.org/W19-3519> (cit. on p. 48).
- [11] D. Budgen and P. Brereton. “Performing systematic literature reviews in software engineering”. In: *Proceedings of the 28th international conference on Software engineering*. 2006, pp. 1051–1052 (cit. on p. 17).
- [12] S. Burrell, S. Ruxton, and N. Westmarland. *What are Gender Norms? Why do they matter?* 2020. URL: <https://www.durham.ac.uk/research/institutes-and-centres/research-violence-abuse/research-profile/geo-toolkit/what-are-gender-norms/> (visited on 2024-07-27) (cit. on p. 48).
- [13] CORE Rankings Portal. <https://www.core.edu.au/conference-portal>. Accessed: 2024-02-1 (cit. on p. 25).
- [14] P. Dave. *Google Bars Uses of its Artificial Intelligence Tech in Weapons*. 2018. URL: <https://www.reuters.com/article/us-alphabet-ai/google-bars-uses-of-its-artificial-intelligence-tech-in-weapons-idUSKCN1J32M7/> (visited on 2024-01-06) (cit. on p. 2).
- [15] M. Deng et al. “A privacy threat analysis framework: supporting the elicitation and fulfillment of privacy requirements”. In: *Requirements Engineering* 16.1 (2011), pp. 3–32 (cit. on p. 43).
- [16] M. Dowd, J. McDonald, and J. Schuh. *The art of software security assessment: Identifying and preventing software vulnerabilities*. Pearson Education, 2006 (cit. on p. 42).
- [17] S. Dritsas et al. “A knowledge-based approach to security requirements for e-health applications”. In: *Electronic Journal for E-Commerce Tools and Applications* (2006), pp. 1–24 (cit. on p. 43).
- [18] M. Fernández-López, A. Gomez-Perez, and N. Juristo. “METHONTOLOGY: from ontological art towards ontological engineering”. In: *Engineering Workshop on Ontological Engineering (AAAI97)* (1997-03) (cit. on p. 31).
- [19] D. Firesmith. “Specifying reusable security requirements.” In: *J. Object Technol.* 3.1 (2004), pp. 61–75 (cit. on pp. 27–29, 41–43).
- [20] M. Flanagan, D. C. Howe, and H. Nissenbaum. *Embodying values in technology: Theory and practice*. na, 2008 (cit. on p. 2).
- [21] D. Ford et al. “Paradise unplugged: identifying barriers for female participation on stack overflow”. In: *Proceedings of the 2016 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering*. FSE 2016. Seattle, WA, USA: Association for Computing Machinery, 2016, pp. 846–857. ISBN: 9781450342186.

- DOI: [10.1145/2950290.2950331](https://doi.org/10.1145/2950290.2950331). URL: <https://doi.org/10.1145/2950290.2950331> (cit. on p. 48).
- [22] J. R. Galliers. *A theoretical framework for computer models of cooperative dialogue, acknowledging multi-agent conflict*. Open University (United Kingdom), 1989 (cit. on p. 46).
- [23] E. I. for Gender Equality. *gender equity*. URL: <https://eige.europa.eu/publications-resources/thesaurus/terms/1262> (visited on 2024-09-25) (cit. on p. 40).
- [24] M. Gharib, J. Mylopoulos, and P. Giorgini. “COPri-A core ontology for privacy requirements engineering”. In: *Research Challenges in Information Science: 14th International Conference, RCIS 2020, Limassol, Cyprus, September 23–25, 2020, Proceedings 14*. Springer. 2020, pp. 472–489 (cit. on p. 43).
- [25] J. A. Goguen. “Formality and Informality in Requirements Engineering”. In: *International Conference on Requirements Engineering*. 1996. URL: <https://api.semanticscholar.org/CorpusID:17375925> (cit. on p. 6).
- [26] A. M. González Ramos and T. Rojas-Rajs. “Inclusion of Gender Perspective in Design and IT Environments”. In: *Proceedings of the XVII International Conference on Human Computer Interaction*. 2016, pp. 1–4. DOI: [10.1145/2998626.2998633](https://doi.org/10.1145/2998626.2998633) (cit. on p. 49).
- [27] Google Scholar. <https://scholar.google.com/intl/en/scholar/about.html>. Accessed: 2024-01-24 (cit. on p. 23).
- [28] N. Grant. “Google Worried Israeli Contract Could Enable Human Rights Violations”. In: *The New York Times* (2024-12). Accessed: 2025-01-03. URL: <https://www.nytimes.com/2024/12/03/technology/google-israel-contract-project-nimbus.html> (cit. on p. 2).
- [29] M. J. Hassan. “Practical Challenges for Social Justice: Some Fundamental Socio-Political Issues”. In: *The Academic: International Journal of Multidisciplinary Research* 2.8 (2024) (cit. on p. 36).
- [30] A. Hendriana, T. Raharjo, and A. Nurfitriani. “Approaches in Determining User Story Quality through Requirement Elicitation: A Systematic Literature Review”. In: *Indonesian Journal of Computer Science* 12.6 (2023) (cit. on p. 10).
- [31] F. History and Ourselves. *Exploring the Concept of Identity*. 2021. URL: <https://www.facinghistory.org/resource-library/exploring-concept-identity> (visited on 2024-07-27) (cit. on p. 49).
- [32] G. Hofstede, G. Hofstede, and M. Minkov. *Cultures and Organizations: Software of the Mind, Third Edition*. McGraw Hill LLC, 2010. ISBN: 9780071770156. URL: <https://books.google.pt/books?id=o40qTgV3V00C> (cit. on p. 46).

- [33] J. Hreha. *What is Social Environment In Behavioral Science?* URL: <https://www.thebehavioralscientist.com/glossary/social-environment> (visited on 2024-07-13) (cit. on p. 45).
- [34] W. Hussain et al. "Human Values in Software Engineering: Contrasting Case Studies of Practice". In: *IEEE Transactions on Software Engineering* 48.5 (2022), pp. 1818–1833. DOI: [10.1109/TSE.2020.3038802](https://doi.org/10.1109/TSE.2020.3038802) (cit. on pp. 11, 12, 15).
- [35] *Independent: Babies born out of wedlock abandoned on Saudi streets due to fears of punishment, campaigners warn.* <https://www.independent.co.uk/news/world/middle-east/saudi-arabia-born-babies-streets-abortion-marriage-wedlock-a8867571.html>. Accessed: 2024-01-22 (cit. on p. 11).
- [36] S. F. R. Institute. *What is social cohesion?* URL: <https://scanloninstitute.org.au/research/mapping-social-cohesion/what-social-cohesion> (visited on 2024-07-18) (cit. on p. 37).
- [37] M. Kay, C. Matuszek, and S. A. Munson. "Unequal Representation and Gender Stereotypes in Image Search Results for Occupations". In: *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*. CHI '15. Seoul, Republic of Korea: Association for Computing Machinery, 2015, pp. 3819–3828. ISBN: 9781450331456. DOI: [10.1145/2702123.2702520](https://doi.org/10.1145/2702123.2702520). URL: <https://doi.org/10.1145/2702123.2702520> (cit. on p. 48).
- [38] B. Kitchenham and S. Charters. "Guidelines for performing Systematic Literature Reviews in Software Engineering". In: 2 (2007-01) (cit. on p. 18).
- [39] kiteworks. *Security Misconfiguration Vulnerabilities: Risks, Impacts, and Prevention.* URL: <https://www.kiteworks.com/risk-compliance-glossary/security-misconfigurations/> (visited on 2024-08-31) (cit. on p. 42).
- [40] M. Krumbholz and N. Maiden. "The implementation of enterprise resource planning packages in different organisational and national cultures". In: *Information systems* 26.3 (2001), pp. 185–204 (cit. on pp. 27–29, 44–46).
- [41] P. Lago et al. "Framing sustainability as a property of software quality". In: *Communications of the ACM* 58.10 (2015), pp. 70–78 (cit. on p. 37).
- [42] C. Landorf. "Evaluating social sustainability in historic urban environments". In: *International Journal of Heritage Studies* 17.5 (2011), pp. 463–477 (cit. on pp. 27–29, 37).
- [43] S. Leavy. "Gender bias in artificial intelligence: the need for diversity and gender theory in machine learning". In: *Proceedings of the 1st International Workshop on Gender Equality in Software Engineering*. GE '18. Gothenburg, Sweden: Association for Computing Machinery, 2018, pp. 14–16. ISBN: 9781450357388. DOI: [10.1145/3195570.3195580](https://doi.org/10.1145/3195570.3195580). URL: <https://doi.org/10.1145/3195570.3195580> (cit. on p. 49).

- [44] B. Lenz, V. Kolarova, and K. Stark. “Gender Issues in the Digitalized ‘Smart’ Mobility World – Conceptualization and Empirical Findings Applying a Mixed Methods Approach”. In: *HCI in Mobility, Transport, and Automotive Systems*. Ed. by H. Krömker. Cham: Springer International Publishing, 2019, pp. 378–392. ISBN: 978-3-030-22666-4 (cit. on p. 49).
- [45] L. Livera. *Top 50 Common Types of Cybersecurity Attacks: A Comprehensive Guide*. 2024. URL: <https://www.linkedin.com/pulse/top-50-common-types-cybersecurity-attacks-guide-lahiru-livera-ndcwc> (visited on 2024-08-31) (cit. on p. 42).
- [46] J. M. Lourenço. *The NOVAthesis L^AT_EX Template User’s Manual*. NOVA University Lisbon. 2021. URL: <https://github.com/joaomlourenco/novathesis/raw/main/template.pdf> (cit. on p. i).
- [47] N. A. M. Maiden. “CREWS-SAVRE: Scenarios for acquiring and validating requirements”. In: *Domain Modelling for Interactive Systems Design* (1998), pp. 39–66 (cit. on p. 45).
- [48] A. Marczak-Czajka, J. Nabrzyski, and J. Cleland-Huang. “Accountable Design for Individual, Societal, and Regulated Values in the UAV Domain”. In: *2023 IEEE 31st International Requirements Engineering Conference (RE)*. 2023, pp. 287–292. DOI: [10.1109/RE57278.2023.00037](https://doi.org/10.1109/RE57278.2023.00037) (cit. on pp. 27–29, 38, 44).
- [49] S. McKenzie. “Adult and vocational education for social sustainability: A new concept for tvet for sustainable development”. In: *Work, Learning and Sustainable Development: Opportunities and Challenges*. Springer, 2009, pp. 177–186 (cit. on p. 36).
- [50] A. Meijer. “Transparency”. In: (2014) (cit. on p. 44).
- [51] Merriam-Webster. *culture*. URL: <https://www.merriam-webster.com/dictionary/culture> (visited on 2024-07-13) (cit. on p. 44).
- [52] D. Miranda, J. Araújo, and G. Liebel. “A Conceptual Model For Web Accessibility Requirements In Agile Development”. In: *Proceedings of the 1st IEEE/ACM Workshop on Multi-Disciplinary, Open, and RElevant Requirements Engineering*. MO2RE 2024. Lisbon, Portugal: Association for Computing Machinery, 2024, pp. 15–21. ISBN: 9798400705694. DOI: [10.1145/3643666.3648580](https://doi.org/10.1145/3643666.3648580). URL: <https://doi.org/10.1145/3643666.3648580> (cit. on pp. 27–29, 40).
- [53] S. Monteiro. *Culture and influence in the workplace: Individualism vs Collectivism*. URL: <https://www.peoplehum.com/blog/culture-and-influence-in-the-workplace-collectivism-individualism> (visited on 2024-09-20) (cit. on p. 45).

- [54] D. Mougouei et al. "Operationalizing Human Values in Software: A Research Roadmap". In: *Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. ESEC/FSE 2018. Lake Buena Vista, FL, USA: Association for Computing Machinery, 2018, pp. 780–784. ISBN: 9781450355735. DOI: [10.1145/3236024.3264843](https://doi.org/10.1145/3236024.3264843). URL: <https://doi.org/10.1145/3236024.3264843> (cit. on p. 2).
- [55] nidirect. *Human rights in the workplace*. URL: <https://www.nidirect.gov.uk/articles/human-rights-workplace> (visited on 2024-07-18) (cit. on p. 38).
- [56] N. Noy and D. McGuinness. "Ontology Development 101: A Guide to Creating Your First Ontology". In: *Knowledge Systems Laboratory* 32 (2001-01) (cit. on p. 31).
- [57] I. Nunes, A. Moreira, and J. Araujo. "Gire: Gender-inclusive requirements engineering". In: *Data & Knowledge Engineering* 143 (2023), p. 102108 (cit. on pp. 27–29, 39, 47–52).
- [58] B. Nuseibeh and S. Easterbrook. "Requirements engineering: a roadmap". In: *Proceedings of the Conference on the Future of Software Engineering*. 2000, pp. 35–46 (cit. on p. 6).
- [59] O. J. Okesola et al. "Qualitative comparisons of elicitation techniques in requirement engineering". In: *ARPN J. Eng. Appl. Sci* 14.2 (2019), pp. 565–570 (cit. on p. 10).
- [60] W. H. Organization. *Sexual health*. URL: https://www.who.int/health-topics/sexual-health#tab=tab_2 (visited on 2024-07-27) (cit. on p. 48).
- [61] C. Pacheco, I. García, and M. Reyes. "Requirements elicitation techniques: a systematic literature review based on the maturity of the techniques". In: *IET Software* 12.4 (2018), pp. 365–378. DOI: <https://doi.org/10.1049/iet-sen.2017.0144>. eprint: <https://ietresearch.onlinelibrary.wiley.com/doi/pdf/10.1049/iet-sen.2017.0144>. URL: <https://ietresearch.onlinelibrary.wiley.com/doi/abs/10.1049/iet-sen.2017.0144> (cit. on p. 8).
- [62] M. M. Peixoto et al. "Towards a Catalog of Privacy Related Concepts." In: *REFSQ Workshops*. 2020 (cit. on pp. 27–29, 41–43).
- [63] B. Penzenstadler and H. Femmer. "A Generic Model for Sustainability with Process- and Product-specific Instances". In: *Proceedings of the 2013 workshop on Green in/by software engineering*. 2013-03, pp. 3–8. DOI: [10.1145/2451605.2451609](https://doi.org/10.1145/2451605.2451609) (cit. on pp. 27–29, 37, 39, 43).
- [64] H. Perera et al. "A Study on the Prevalence of Human Values in Software Engineering Publications, 2015 – 2018". In: *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ICSE '20. Seoul, South Korea: Association for Computing Machinery, 2020, pp. 409–420. ISBN: 9781450371216. DOI: [10.1145/33](https://doi.org/10.1145/33)

- 77811.3380393. URL: <https://doi.org/10.1145/3377811.3380393> (cit. on pp. 2, 3).
- [65] K. Petersen, S. Vakkalanka, and L. Kuzniarz. “Guidelines for conducting systematic mapping studies in software engineering: An update”. In: *Information and software technology* 64 (2015), pp. 1–18 (cit. on p. 17).
- [66] K. Petersen et al. “Systematic mapping studies in software engineering”. In: *12th International Conference on Evaluation and Assessment in Software Engineering (EASE)* 12. 2008, pp. 1–10 (cit. on pp. 17–19).
- [67] A. Porter. *Privacy Engineering: Safeguarding Data in Today’s World*. 2023. URL: <https://bigid.com/blog/what-is-privacy-engineering/> (visited on 2024-08-31) (cit. on p. 43).
- [68] C. U. Press. *belief*. URL: <https://dictionary.cambridge.org/pt/dicionario/ingles/belief> (visited on 2024-09-20) (cit. on p. 46).
- [69] C. U. Press. *broad-minded*. URL: <https://dictionary.cambridge.org/dictionary/english/broad-minded> (visited on 2024-09-24) (cit. on p. 40).
- [70] C. U. Press. *community*. URL: <https://dictionary.cambridge.org/dictionary/english/community> (visited on 2024-09-22) (cit. on p. 37).
- [71] C. U. Press. *disability*. URL: <https://dictionary.cambridge.org/dictionary/english/disability> (visited on 2024-09-24) (cit. on p. 40).
- [72] C. U. Press. *discrimination*. URL: <https://dictionary.cambridge.org/dictionary/english/discrimination> (visited on 2024-09-24) (cit. on p. 38).
- [73] C. U. Press. *education*. URL: <https://dictionary.cambridge.org/dictionary/english/education> (visited on 2024-09-24) (cit. on p. 39).
- [74] C. U. Press. *equality*. URL: <https://dictionary.cambridge.org/dictionary/english/equality> (visited on 2024-09-22) (cit. on p. 36).
- [75] C. U. Press. *knowledge*. URL: <https://dictionary.cambridge.org/dictionary/english/knowledge> (visited on 2024-09-24) (cit. on p. 39).
- [76] C. U. Press. *social-justice*. URL: <https://dictionary.cambridge.org/dictionary/english/social-justice> (visited on 2024-09-22) (cit. on p. 36).
- [77] C. U. Press. *wisdom*. URL: <https://dictionary.cambridge.org/dictionary/english/wisdom> (visited on 2024-09-24) (cit. on p. 40).
- [78] U. N. W. F. Programme. *Freerice*. URL: <https://freerice.com/> (visited on 2024-09-26) (cit. on p. 63).
- [79] U. N. W. F. Programme. *ShareTheMeal*. URL: <https://sharethemeal.org/en-us/> (visited on 2024-07-27) (cit. on p. 36).

-
- [80] J. A. Rode. "A theoretical agenda for feminist HCI". In: *Interacting with Computers* 23.5 (2011). Feminism and HCI: New Perspectives, pp. 393–400. ISSN: 0953-5438. DOI: <https://doi.org/10.1016/j.intcom.2011.04.005>. URL: <https://www.sciencedirect.com/science/article/pii/S0953543811000385> (cit. on pp. 48, 49).
 - [81] M. Rokeach. *The nature of human values*. Free press, 1973 (cit. on p. 11).
 - [82] T. R. D. Saputri and S.-W. Lee. "Addressing sustainability in the requirements engineering process: From elicitation to functional decomposition". In: *Journal of Software: Evolution and Process* 32.8 (2020), e2254 (cit. on p. 47).
 - [83] A. Schlesinger, W. K. Edwards, and R. E. Grinter. "Intersectional HCI: Engaging Identity through Gender, Race, and Class". In: *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. CHI '17. Denver, Colorado, USA: Association for Computing Machinery, 2017, pp. 5412–5427. ISBN: 9781450346559. DOI: [10.1145/3025453.3025766](https://doi.org/10.1145/3025453.3025766). URL: <https://doi.org/10.1145/3025453.3025766> (cit. on p. 49).
 - [84] E.-M. Schön, J. Thomaschewski, and M. J. Escalona. "Agile Requirements Engineering: A systematic literature review". In: *Computer standards & interfaces* 49 (2017), pp. 79–91 (cit. on p. 6).
 - [85] S. Schwartz. "Basic Human Values: An Overview". In: (2005-01) (cit. on pp. 1, 12–14, 46).
 - [86] S. Schwartz. "Universals in the Content and Structure of Values: Theoretical Advances and Empirical Tests in 20 Countries". In: *Advances in Experimental Social Psychology* 25 (1992-12), pp. 1–65. DOI: [10.1016/S0065-2601\(08\)60281-6](https://doi.org/10.1016/S0065-2601(08)60281-6) (cit. on pp. 2, 3, 12–14, 16).
 - [87] M. Shahin et al. "Operationalizing Human Values in Software Engineering: A Survey". In: *IEEE Access* 10 (2022), pp. 75269–75295. DOI: [10.1109/ACCESS.2022.3190975](https://doi.org/10.1109/ACCESS.2022.3190975) (cit. on p. 12).
 - [88] S. Sharma and S. K. Pandey. "Article: Revisiting Requirements Elicitation Techniques". In: *International Journal of Computer Applications* 75.12 (2013-08). Full text available, pp. 35–39 (cit. on pp. 6, 8–10).
 - [89] S. Sharma and S. K. Pandey. "Requirements elicitation: Issues and challenges". In: *2014 International Conference on Computing for Sustainable Global Development (INDIACom)*. 2014, pp. 151–155. DOI: [10.1109/IndiaCom.2014.6828119](https://doi.org/10.1109/IndiaCom.2014.6828119) (cit. on pp. 8, 9).
 - [90] A. P. Silva and A. F. Frère. "Virtual environment to quantify the influence of colour stimuli on the performance of tasks requiring attention". In: *Biomedical engineering online* 10 (2011), pp. 1–14 (cit. on p. 16).

- [91] Y. Singh and R. Malhotra. *Object-oriented software engineering*. PHI Learning Pvt. Ltd., 2012 (cit. on p. 11).
- [92] K. Spiel, O. L. Haimson, and D. Lottridge. “How to do better with gender on surveys: a guide for HCI researchers”. In: *Interactions* 26.4 (2019-06), pp. 62–65. ISSN: 1072-5520. DOI: [10.1145/3338283](https://doi.org/10.1145/3338283). URL: <https://doi.org/10.1145/3338283> (cit. on p. 49).
- [93] *The National News: The rules are clear, says lawyer: no kissing allowed in Dubai*. <https://www.thenationalnews.com/uae/heritage/2023/03/06/the-rules-are-clear-says-lawyer-no-kissing-allowed-in-dubai/>. Accessed: 2024-01-22 (cit. on p. 11).
- [94] S. Tiwari and S. S. Rathore. “A methodology for the selection of requirement elicitation techniques”. In: *arXiv preprint arXiv:1709.08481* (2017) (cit. on pp. 9, 10).
- [95] UNESCO. *Promoting Gender Equity in and through Education*. URL: <https://www.ibe.unesco.org/en/articles/promoting-gender-equity-and-through-education> (visited on 2024-09-25) (cit. on p. 40).
- [96] UNESCO. *The Right to Education*. URL: <https://www.unesco.org/en/right-education> (visited on 2024-09-25) (cit. on p. 39).
- [97] O. of the United Nations High Commissioner for Human Rights. *Gender Stereotyping*. URL: <https://www.ohchr.org/en/women/gender-stereotyping> (visited on 2024-07-27) (cit. on pp. 48, 49).
- [98] M. University. *Definitive Guide to Gender Inclusion*. URL: <https://online.maryville.edu/online-bachelors-degrees/liberal-studies/guide-to-gender-inclusion/> (visited on 2024-07-27) (cit. on p. 48).
- [99] B. University of California. *Types of Assistive Technology*. URL: <https://dap.berkeley.edu/types-assistive-technology> (visited on 2024-09-25) (cit. on p. 40).
- [100] G. Van Blarckom, J. J. Borking, and J. E. Olk. “Handbook of privacy and privacy-enhancing technologies”. In: *Privacy Incorporated Software Agent (PISA) Consortium, The Hague* 198 (2003), p. 14 (cit. on p. 43).
- [101] A. Van Lamsweerde. “Goal-oriented requirements engineering: A guided tour”. In: *Proceedings fifth ieee international symposium on requirements engineering*. IEEE. 2001, pp. 249–262 (cit. on p. 11).
- [102] M. West, R. Kraut, and H. Ei Chew. “I’d blush if I could: closing gender divides in digital skills through education”. In: (2019) (cit. on p. 48).
- [103] G. Williams. “Are you sure your software is gender-neutral?” In: *Interactions* 21.1 (2014-01), pp. 36–39. ISSN: 1072-5520. DOI: [10.1145/2524808](https://doi.org/10.1145/2524808). URL: <https://doi.org/10.1145/2524808> (cit. on p. 49).

- [104] T. Winkler. "Human Values as the Basis for Sustainable Software Development". In: *2018 IEEE International Symposium on Technology and Society (ISTAS)*. IEEE. 2018, pp. 37–42 (cit. on pp. 27–29).
- [105] WithSecure. *Cybersecurity Unveiled: Understanding External and Internal Threats*. 2023. URL: <https://www.withsecure.com/en/expertise/blog-posts/external-and-internal-threats> (visited on 2024-08-31) (cit. on p. 42).
- [106] C. Wohlin et al. *Experimentation in software engineering*. Springer Science & Business Media, 2012 (cit. on p. 94).
- [107] P. Zave. "Classification of research efforts in requirements engineering". In: *ACM Comput. Surv.* 29.4 (1997-12), pp. 315–321. ISSN: 0360-0300. DOI: [10.1145/267580.267581](https://doi.org/10.1145/267580.267581). URL: <https://doi.org/10.1145/267580.267581> (cit. on p. 6).
- [108] D. Zowghi and C. Coulin. "Requirements Elicitation: A Survey of Techniques, Approaches, and Tools". In: *Engineering and Managing Software Requirements*. Ed. by A. Aurum and C. Wohlin. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 19–46. ISBN: 978-3-540-28244-0. DOI: [10.1007/3-540-28244-0_2](https://doi.org/10.1007/3-540-28244-0_2). URL: https://doi.org/10.1007/3-540-28244-0_2 (cit. on pp. 7, 10).

A

EVALUATION QUESTIONNAIRE

An Approach For Human Values Requirements: The Case Of Universalism Questionnaire

The goal of this questionnaire is to evaluate a conceptual model on universalism requirements (e.g., equality and social justice requirements) which are relevant nowadays for software applications and the respective prototype tool. The conceptual model is a starting point for supporting the elicitation of universalism requirements. This was developed in the context of the MSc dissertation in Computer Science and Engineering of Alexandra Lélé, a student at NOVA School of Science of Technology.

The link to access the tool is available here:

<https://alexaserra0.github.io/universalism-tool/>

You will find the guide to the support tool in the following link:

<https://drive.google.com/file/d/10UIOUywetBr8GZzYyOjGzDPEoREjbXTy/view?usp=sharing>

Alternatively, here is a video explaining how to use the tool:

<https://drive.google.com/file/d/1XfbsnLfMmFDZXusq4BCzLLvPM9krFHVR/view?usp=sharing>

Please read the guide or watch the video before answering the questionnaire.

The following framework document complements the understanding of the model and the tool. (optional)

Link: <https://drive.google.com/file/d/1qHvrjSgO8xbhHY295QotZF4clsfno2-R/view?usp=sharing>

Your answers are anonymous, and any personal information will be kept strictly confidential. The final MSc dissertation will present anonymized answers.

This questionnaire will take about 30 minutes to answer. If you are interested in the results of this work, please contact me using the email below.

Thank you for taking the time to participate in this survey.
We greatly appreciate your help and availability.

Alexandra Lélé (a.lele@campus.fct.unl.pt)

* Indica uma pergunta obrigatória

1. What is your gender? *

Marcar apenas uma oval.

☐ Female

☐ Male

☐ Prefer not to disclose

☐ Outra: _____

2. What is your age? *

3. What is your level of academic training? *

Marcar apenas uma oval.

☐ Bachelor

☐ Master's Degree

☐ PhD

☐ Outra: _____

4. What is the scientific area of your degree? *

E.g., Computer Science

5. What is your current occupation? *

Marcar apenas uma oval.

- ☐ Student (e.g., MSc, BSc, PhD, etc.)
- ☐ Practitioner (e.g., programmer, software engineer, analyst, etc.)
- ☐ Academic (e.g., Professor, lecturer, etc.)
- ☐ Researcher (e.g., Post-doc, research assistant, etc.)
- ☐ Outra: _____

6. What is your experience with conceptual modeling? *

Marcar apenas uma oval.

- ☐ 0-2 years
- ☐ 3-5 years
- ☐ 6-10 years
- ☐ 10+ years

7. In your opinion, how important are universalism values - such as equality, social justice - during software development? *

Marcar apenas uma oval.

	1	2	3	4	5	
Not	☐	☐	☐	☐	☐	Very Important

Overview of the model

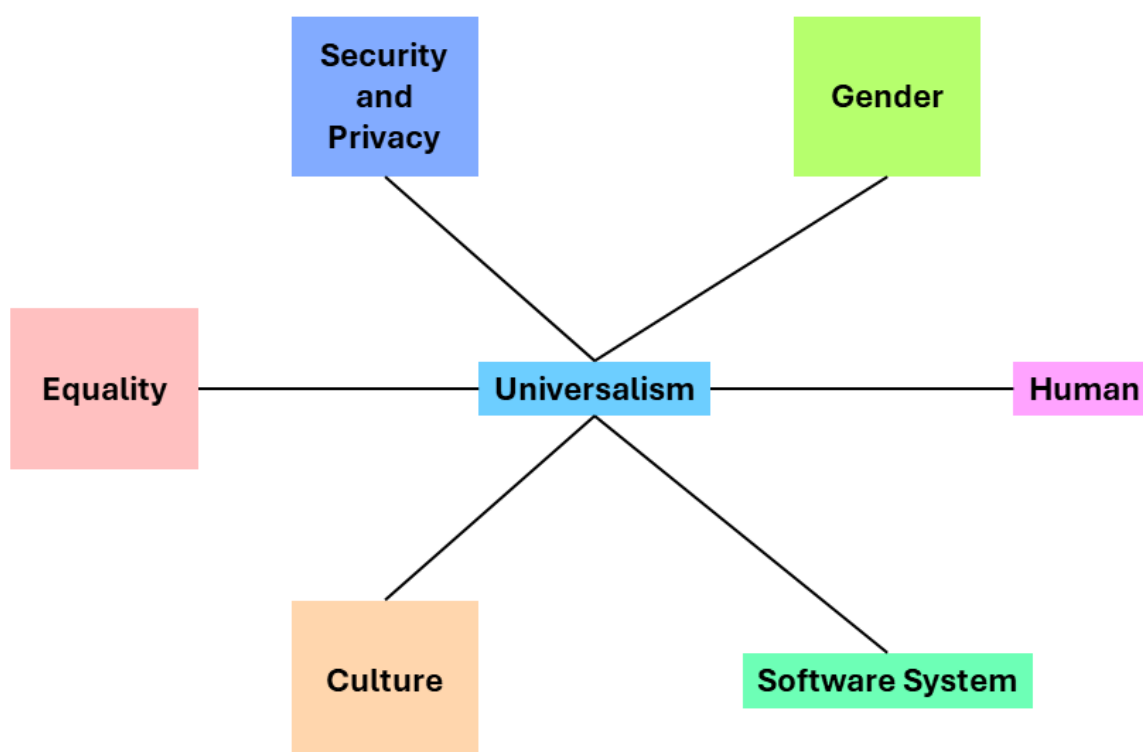
Below you have an image of the conceptual model whose main objective is to consider universalism values when developing software systems. The software system that is created should be inclusive, fair, and secure for every user.

The conceptual model highlights six parts: Equality, Security and Privacy, Gender, Culture, Human, and Software System. In the next sections, the questions refer to the parts of this model.

Due to the size of the model, please open the image available in the link below to view the entire model.

https://drive.google.com/file/d/1iF6L_crMB0hTqlMaFg75tLC0GpAzEgzk/view?usp=sharing

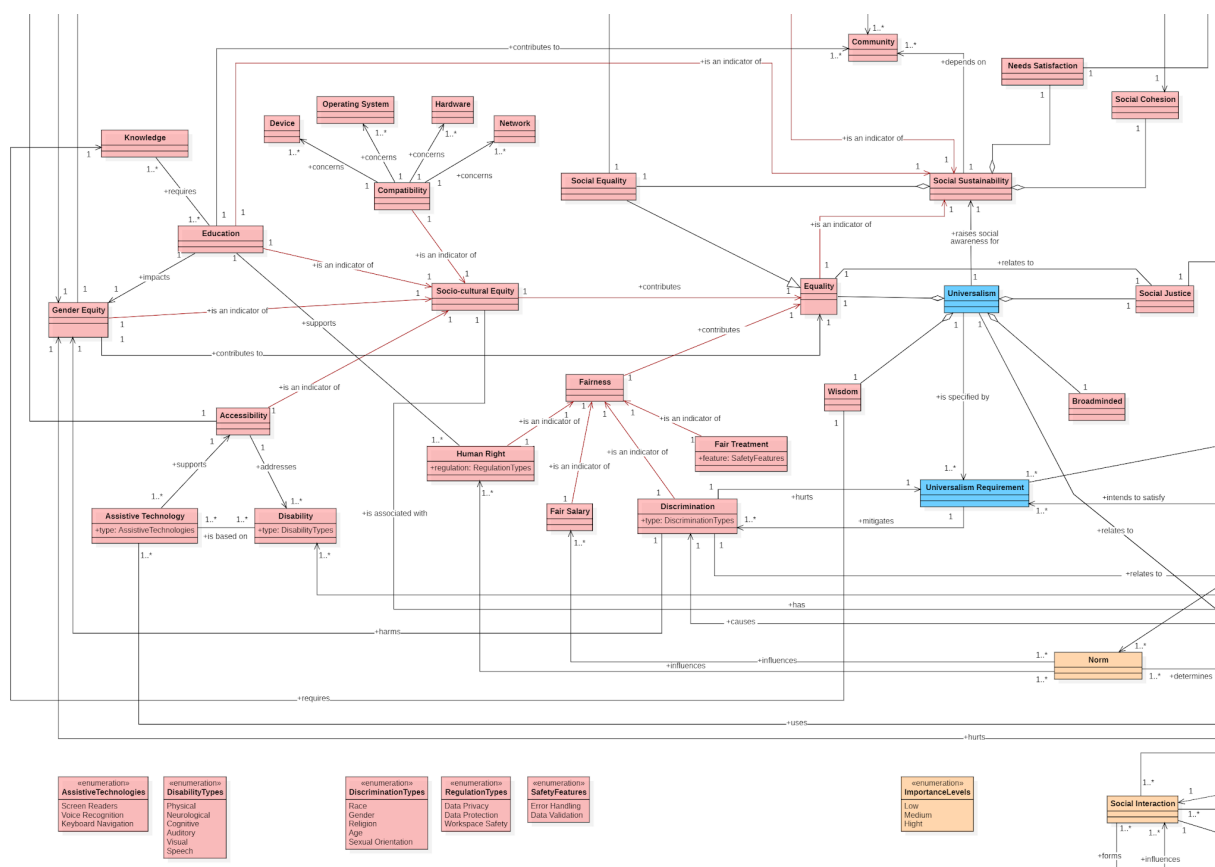
Conceptual Model Overview



Equality

The goal of the equality part is to promote equal access and opportunities for every user in the system, regardless of their background or abilities.

Image Link: https://drive.google.com/file/d/1DZcaPvPYR1hZGOq7ph_7Krx1KjuYTEP/view?usp=sharing



10. Does this model properly include all the necessary concepts and relations regarding Equality?

★

Marcar apenas uma oval.

1 2 3 4 5

Stro ☐ ☐ ☐ ☐ ☐ Strongly agree

The goal of the culture part is to ensure software systems integrate users' beliefs, values, and norms to improve user experience and encourage positive social interactions.

12. Does this model properly include all the necessary concepts and relations regarding Culture? *

Marcar apenas uma oval.

1 2 3 4 5

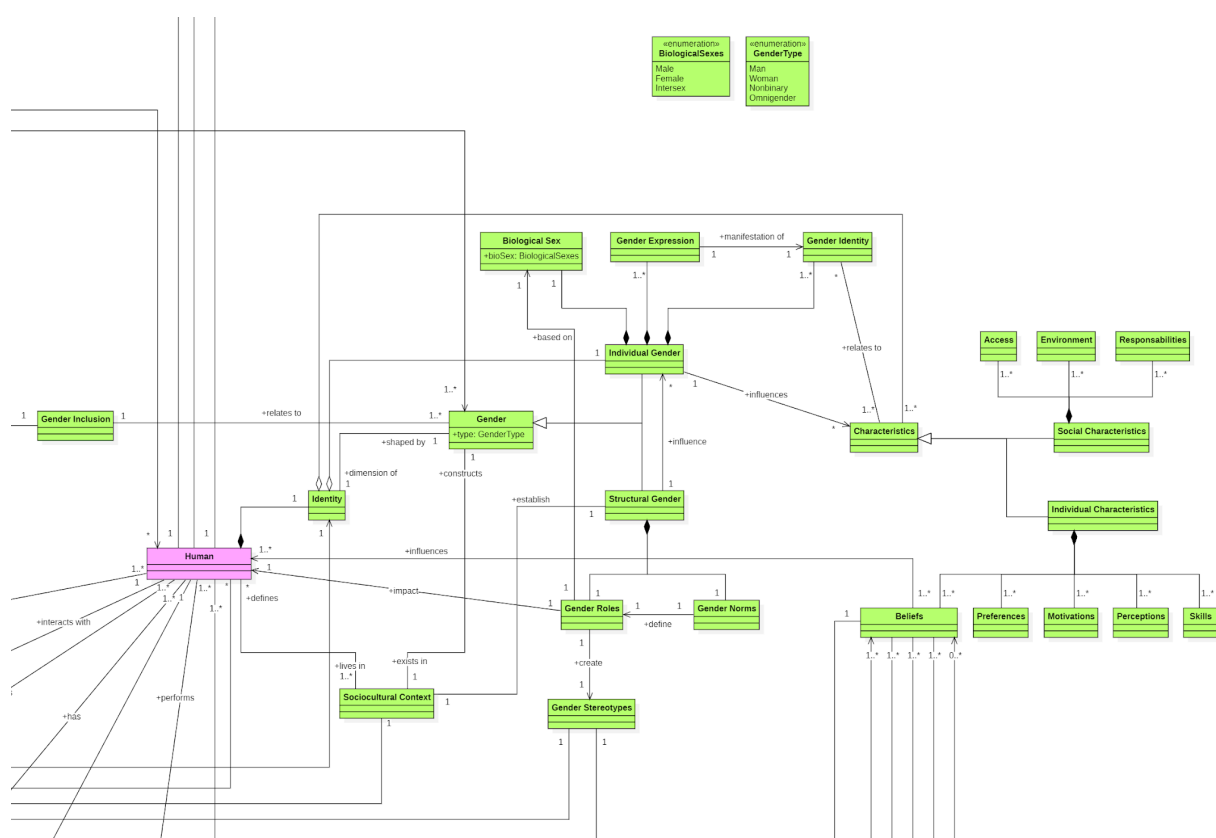
Stro ☐ ☐ ☐ ☐ ☐ Strongly agree

13. If you disagree, please list the concepts (or relationships) you would change or add:

Gender

The goal of the gender part is to make sure that the system is inclusive and respects gender diversity, thus adapting to users' gender identities while not discriminating against them.

Image Link: <https://drive.google.com/file/d/14ICVu6uH8ePDI8J8-G1hLqJjsbMfnnJ-/view?usp=sharing>



14. Does this model properly include all the necessary concepts and relations regarding Gender?

★

Marcar apenas uma oval.

1 2 3 4 5

Stro ☐ ☐ ☐ ☐ ☐ Strongly agree

15. If you disagree, please list the concepts (or relationships) you would change or add:

Software System

The software system part represents the software system that intends to satisfy universalism requirements.

Please open the image in the link below to view the entire model.

https://drive.google.com/file/d/1iF6L_crMB0hTqlMaFg75tLC0GpAzEgzk/view?usp=sharing

16. Does this model properly include all the necessary concepts and relations regarding Software System? *

Marcar apenas uma oval.

1 2 3 4 5

Stro ☐ ☐ ☐ ☐ ☐ Strongly agree

17. If you disagree, please list the concepts (or relationships) you would change or add:

Human

The human part represents the individuals involved with a software system.

Please open the image in the link below to view the entire model.

https://drive.google.com/file/d/1iF6L_crMB0hTqIMaFg75tLC0GpAzEgzk/view?usp=sharing

18. Does this model properly include all the necessary concepts and relations regarding Human? *

Marcar apenas uma oval.

1 2 3 4 5

Stro ☐ ☐ ☐ ☐ ☐ Strongly agree

19. If you disagree, please list the concepts (or relationships) you would change or add:

General opinion about the conceptual model

For each of the following questions, please provide the answer that better reflects your opinion about this model.

Please open the image in the link below to view the entire model.

https://drive.google.com/file/d/1iF6L_crMB0hTqIMaFg75tLC0GpAzEgzk/view?usp=sharing

20. How useful is this conceptual model? *

Marcar apenas uma oval.

1 2 3 4 5

Not ☐ ☐ ☐ ☐ ☐ Very useful

21. How difficult is it to understand this conceptual model? *

Marcar apenas uma oval.

1 2 3 4 5

Not ☐ ☐ ☐ ☐ ☐ Very difficult

22. Would you use this conceptual model to develop an application where universalism requirements are relevant? *

Marcar apenas uma oval.

1 2 3 4 5

Stro ☐ ☐ ☐ ☐ ☐ Strongly agree

23. Would you recommend this conceptual model? *

Marcar apenas uma oval.

1 2 3 4 5

Don' ☐ ☐ ☐ ☐ ☐ Strongly recommend

24. What are the strongest points about this conceptual model?

25. What are the weakest points of this conceptual model?

Prototype Evaluation

Please choose the option that better suits your opinion about the the tool developed.

The link to access the tool is available here:

<https://alexaserra0.github.io/universalism-tool/>

You will find the guide to the support tool in the following link:

<https://drive.google.com/file/d/10UIOUywetBr8GZzYyOjGzDPEoREjbXTy/view?usp=sharing>

Alternatively, here is a video explaining how to use the tool:

<https://drive.google.com/file/d/1XfbsnLfMmFDZXusq4BCzLLvPM9krFHVR/view?usp=sharing>

Please read the guide or watch the video before answering the questionnaire.

Tool page

The following questions are about the tool page, the one that follows the "START" button and presents the model.

26. About the tool page, did you have any difficulty in exploring the model? *

Marcar apenas uma oval.

1 2 3 4 5

Very ☐ ☐ ☐ ☐ ☐ Very easy

27. In case you had any difficulty, please describe the problem.

28. About the tool page, did you have any difficulty in the selection and saving of the concepts? *

Marcar apenas uma oval.

1 2 3 4 5

Very ☐ ☐ ☐ ☐ ☐ Very easy

29. In case you had any difficulty, please describe the problem.

Questions Page

The following questions are about the questions page, the one that follows the "NEXT" button and presents the questions of the concepts.

30. About the questions page, did you have any difficulty in selecting and removing *
the concepts?

Marcar apenas uma oval.

	1	2	3	4	5	
Very	☐	☐	☐	☐	☐	Very easy

31. In case you had any difficulty, please describe the problem.

32. Was the case example of the app ShareTheMeal useful? *

Marcar apenas uma oval.

	1	2	3	4	5	
Not	☐	☐	☐	☐	☐	Very useful

33. In case you didn't find it useful, please explain why.

Document Page

The following questions are about the document page, the one that follows the "NEXT" button and is the final step in saving the document.

34. About the document page, did you have any difficulty in adding and removing participants? *

Marcar apenas uma oval.

1 2 3 4 5

Very ☐ ☐ ☐ ☐ ☐ Very easy

35. In case you had any difficulty, please describe the problem.

Documents Page

The following questions are about the documents page, the one that follows the "SAVE DOCUMENT" button and presents a list of all documents.

36. About the documents page, did you have any difficulty in exporting, editing or deleting a document? *

Marcar apenas uma oval.

1 2 3 4 5

Very ☐ ☐ ☐ ☐ ☐ Very easy

37. In case you had any difficulty, please describe the problem.

General opinion about the tool

38. How intuitive would you rate the interface? *

Marcar apenas uma oval.

1 2 3 4 5

Very ☐ ☐ ☐ ☐ ☐ Very intuitive

39. How is the navigation of the tool? *

Marcar apenas uma oval.

1 2 3 4 5

Very ☐ ☐ ☐ ☐ ☐ Very easy

40. Do you have any suggestions on how we could improve the tool?

Framework Document

41. Was the document useful for your understanding of the model? *

Marcar apenas uma oval.

1 2 3 4 5

Not ☐ ☐ ☐ ☐ ☐ Very useful

42. In general, was the description of the concepts comprehensible? *

Marcar apenas uma oval.

1 2 3 4 5

Not ☐ ☐ ☐ ☐ ☐ Very comprehensible

Guide Evaluation

Only answer if you read the guide.

43. Was the guide useful?

Marcar apenas uma oval.

	1	2	3	4	5	
Not	☐	☐	☐	☐	☐	Very useful

44. Was the guide easy to understand?

Marcar apenas uma oval.

	1	2	3	4	5	
Not	☐	☐	☐	☐	☐	Very useful

Video Evaluation

Only answer if you watched the video.

45. Was the video useful?

Marcar apenas uma oval.

	1	2	3	4	5	
Not	☐	☐	☐	☐	☐	Very useful

Este conteúdo não foi criado nem aprovado pela Google.

Google Formulários





2024 An Approach for Human Values Requirements: The Case of Universalism Alexandra Lélée

