UNIVERSITY OF TARTU
Faculty of Science and Technology
Institute of Computer Science
Computer Science Curriculum

Elvin Mirzazada

Towards a Novel Taxonomy for Requirements Interdependencies

Master's Thesis (30 ECTS)

Supervisor(s): Mohamad Gharib, Ph.D.

Tartu 2024

Towards a Novel Taxonomy for Requirements Interdependencies

Abstract: Requirements interdependencies capture how requirements relate to and affect each other, and it is one of the main problems to be handled for delivering quality requirements, and in turn, software systems with high quality. This said, requirements interdependencies cannot be overlooked or ignored and must be properly handled since requirements dependencies influence several design, development, and implementation decisions, and inappropriate handling of such interdependencies can lead to software system development failures. Although various interdependencies among requirements have been considered in the literature (e.g., requires, refines, similar, or conflicts), they are not able to cope with the advancement on the requirements side. More specifically, systems are becoming more complex, leading to more complex interdependencies among their requirements, which available interdependencies might not be able to capture. This thesis aims to solve this problem by developing a new taxonomy of requirements interdependencies that can better understand software requirements and the dependencies between them. The taxonomy aims to overcome the limitations in existing work by proposing a taxonomy that offers a comprehensive set of requirements interdependencies. To achieve that the taxonomy has been mined via a Systematic Literature Review (SLR). The new taxonomy aims to facilitate the production of a more elaborated and expressive set of software system requirements, which will positively contribute to the development of high-quality software systems.

As a result, this thesis aims to provide a solution to a problem encountered in the field of software development, to make requirements analysis more effective and efficient, and to contribute to the production of higher-quality software.

This thesis examined the relationship between software requirements and dependencies in-depth and identified 16 different types of relationships. These relationships are classified into different categories. This thesis offers suggestions for future research to address issues such as expanding the application areas of the taxonomy, including new dependency types and developing automation tools.

Keywords:
Requirements, interdependency, taxonomy, relationship, relation

CERCS: P170, Computer science, numerical analysis, systems, control

Uue nõuete vastastikuste sõltuvuste taksonoomia suunas

Lühikokkuvõte: Nõuete vastastikused sõltuvused kirjeldavad, kuidas nõuded omavahel seotud on ja üksteist mõjutavad. See on üks peamisi probleeme, mida tuleb käsitleda kvaliteetsete nõuete ja seeläbi ka kõrge kvaliteediga tarkvarasüsteemide tarnimisel. Seetõttu ei saa nõuete vastastikuseid sõltuvusi tähelepanuta jätta ega ignoreerida, vaid neid tuleb korralikult hallata, kuna nõuete sõltuvused mõjutavad mitmeid projekteerimis-, arendus- ja juurutamisotsuseid. Selliste sõltuvuste ebaõige käsitlemine võib viia tarkvarasüsteemi arendus ebaõnnestumiseni. Kuigi kirjanduses on käsitletud erinevaid nõuete vastastikuseid sõltuvusi (nt vajab, täpsustab, sarnaneb või on vastuolus), ei suuda need arvestada nõuete valdkonna arengutega. Täpsemalt, süsteemid muutuvad keerukamaks, mis viib keerukamate nõuete vastastikuste sõltuvusteni, mida olemasolevad sõltuvused ei pruugi suuta tabada. Käesoleva väitekirja eesmärk on lahendada see probleem, arendades välja uue nõuete sõltuvuste taksonoomia, mis suudaks paremini mõista tarkvaranõudeid ja nende vahelisi sõltuvusi. Taksonoomia eesmärk on ületada olemasolevate tööde piirangud, pakkudes terviklikumat nõuete vastastikuste sõltuvuste komplekti. Selle saavutamiseks on taksonoomia välja töötatud süsteemse kirjanduse ülevaate (SLR) abil. Uus taksonoomia on mõeldud hõlbustama tarkvarasüsteemi nõuete koostamist, mis avaldab positiivset mõju kõrge kvaliteediga tarkvarasüsteemide arendamisele.

Kokkuvõttes püüab see väitekiri pakkuda lahendust tarkvaraarenduse valdkonnas esinevale probleemile, muuta nõuete analüüs tõhusamaks ja efektiivsemaks ning aidata kaasa kvaliteetsemate tarkvarade tootmisele.

See väitekiri uuris põhjalikult tarkvaranõuete ja sõltuvuste vahelisi seoseid ning tuvastas 16 erinevat tüüpi seoseid. Need seosed on klassifitseeritud erinevatesse kategooriatesse. Väitekiri pakub ettepanekuid edasisteks uuringuteks, mis käsitlevad selliseid teemasid nagu taksonoomia rakendusalade laiendamine, uute sõltuvustüüpide lisamine ja automatiseerimisvahendite arendamine.

Võtmesõnad:
Nõuded, vastastikune sõltuvus, taksonoomia, seos, suhe

CERCS: P170, Arvutiteadus, arvutusmeetodid, süsteemid, juhtimine (automaatjuhtimisteooria)

Contents

1 Introduction

1.1 Problem statement

It is well acknowledged in the Requirements Engineering (RE) community that most requirements are related to one another [GP05]. According to Carlshamre et al., [CSL+01], only a fifth of the requirements are not related to or influenced by any other requirements. Requirements interdependencies capture how requirements relate to and affect each other [MSMS14], and it is one of the main problems to be handled for delivering high-quality requirements, and in turn, a high-quality software system. This said, requirements interdependencies cannot be overlooked or ignored and must be properly dealt with. In particular, requirements dependencies influence several design, development, and implementation decisions [NPV03], and inappropriate handling of such interdependency can lead to software system development failures [NFRN23]. Although various interdependencies among requirements have been considered in the literature (e.g., requires, refines, similar, or conflicts [Moh16]), they are not able to cope with the advancement on the requirements side. More specifically, systems are becoming more complex, leading to more complex interdependencies among their requirements, which available interdependencies might not be able to capture. To this end, understanding and managing requirements interdependencies is critical since it influences the quality of software systems, their development time, and overall project cost.

On the other hand, different terminologies and definitions describe and classify the interdependencies among requirements [NPV03], making it difficult to compare and synthesize research results. Without a structured taxonomy, it isn't easy to identify, understand and properly use existing types of interdependencies among requirements. This leads to potential oversights in analysis and incorrect requirements analysis. Without a clear understanding of the various types of interdependencies, it isn't easy to develop tools and techniques designed to specifically address them. This prevents the creation of practical solutions to improve the requirements analysis processes.

The thesis will be supported by examining various studies in the field of RE. These studies address issues such as requirements interdependencies, requirements management, and requirements analysis. The new taxonomy can be used to identify and classify interdependencies during the requirements analysis process.

Functional requirements interdependencies often occur when one functionality depends on another to operate correctly. For example, the functionality for managing user profiles depends on the login functionality. Without logging in, a user cannot access or modify their profile information.

Non-functional requirements interdependencies are seen when one non-functional requirement affects the satisfaction of another. For instance: increasing security measures, such as encryption, can impact system performance. Enhanced security protocols might slow down the data processing speed.

Interdependencies between functional and non-functional requirements occur when a functional requirement impacts a non-functional one or vice versa. For instance: The implementation of a search function (functional requirement) must ensure it meets usability standards (non-functional requirement). If the search function is too complex or slow, it impacts the overall usability of the system.

Taxonomy can be used to separate these dependencies into categories such as *"functional"*, *"non-functional"*, *"functional and non-functional"* and *"general"*.

Existing research offers a variety of approaches for capturing interdependencies among requirements, but they are often addressed in a fragmented and non-standardized manner. This leads to a lack of common understanding and terminology among researchers. The thesis aims to develop a common understanding among researchers and practitioners and to produce more effective solutions by providing a standard framework to define and classify different types of interdependencies.

1.2 Objectives of proposed research

The objective of the thesis is to develop a novel taxonomy that will provide a more in-depth and comprehensive understanding of the relationships among requirements interdependencies. This will enable better understanding, analysis capabilities and management concerning complex requirements and interrelationships encountered in modern complex software systems.

This taxonomy covers the complex relationships between different types of requirements (functional, non-functional, user, technical, etc.), the types of dependencies that arise in different contexts (business, architecture, design, development, etc.), the specific dependencies brought by new technologies, and the requirements may include evolution.

1.3 Research Questions (RQs)

To achieve the objectives of this research, I have defined the following Research Questions (RQs) are:

RQ1: What are the existing types of requirements interdependence?

RQ2: How are dependencies between existing requirement types used across different requirement types?

RQ3: What are the coverage limitations of existing studies?

1.4 Contributions of the Thesis

The thesis presents a novel taxonomy for comprehending and overseeing the interdependencies among requirements within the field of RE. The new taxonomy improves mutual comprehension among requirements engineers and stakeholders, making it easier to compare and combine various requirements dependencies that have been found. Taxonomy facilitates the creation of novel tools and procedures, such as automated analysis tools and dependency management strategies, that enhance the management of dependencies. Implementing standardized definitions and analyses enhances the comparability of research findings, hence expediting advancements in the discipline. The pragmatic framework facilitates the management of interdependencies in requirements analysis and management, enhancing both the quality of software and the efficiency of development. In this thesis, the details of the proposed taxonomy for the interdependencies of requirements will be discussed, the comparison will be made with application examples and similar studies, and the potential contributions of this taxonomy to the field of software engineering will be evaluated.

1.5 Structure of the Thesis

The rest of this thesis is organized as follows; Section 2 describes the baseline and state-of-the-art/related works. In Section 3, the research methodology is described. The SLR is presented and its results and findings are discussed in Section 4. Section 5 presents the novel taxonomy for requirements interdependencies, and the evaluation of the taxonomy is discussed in Section 6. The taxonomy is validated by applying it to an example that uses the Motivational Goal Model (MGM) example in Section 7, and a discussion about threats to validity is presented in Section 8. Section 9 presents further works and the limitations of this study, and the thesis is concluded in Section 10.

2 Baseline and State of the art/related work

Defining the requirements of a system is crucial in the field of software engineering to ensure the effective development and deployment of any software project. This section focuses on the fundamental principles and current trends in understanding and establishing software requirements. The topic is divided into two subsections: Baseline and Related Work.

2.1 Baseline

The Baseline part focuses on clarifying the key types of requirements and dependencies that are the foundation of any software requirements specification (SRS). Requirements are the fundamental characteristics and capabilities that the program must demonstrate to fulfil the requirements and expectations of stakeholders.

Requirement Types
As per PMBOK (Project Management Body of Knowledge) [Bel07, Wie03], the main types of requirements talked about are (shown in Figure 1):

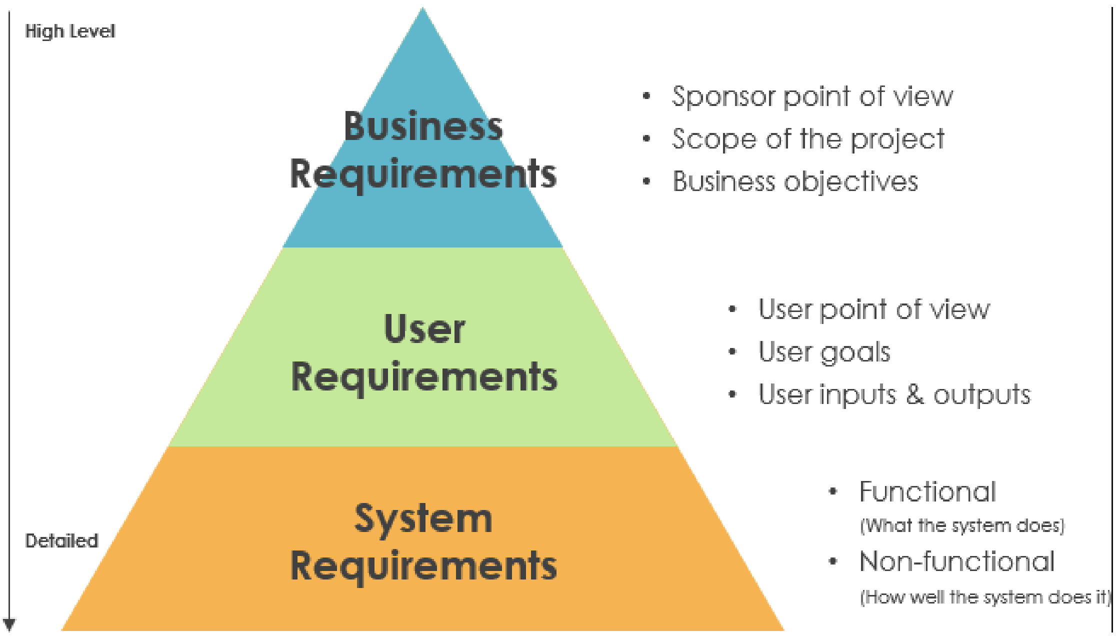

Figure 1. The type of requirements.

Functional Requirements: These requirements specify what the system must accomplish [AAN15, HGC21, SJ05, IB01]. For example, in an e-commerce platform, a functional requirement might state that "customers must be able to search and purchase products."

Non-Functional Requirements: These stipulate how the system should operate, encompassing aspects like performance, reliability, availability, and security [SAC19, MSMS14, HGC21, SHSM14, JH08]. Examples include "the system must respond within 1 second" or "the system must be secure against unauthorized access."

User Requirements: These requirements reflect the expectations of users regarding system interaction [MAL+20]. An example is that "a user should be able to easily add an item to their cart."

Business Requirements: These requirements align the system's functionality with the organization's business objectives [GG19, MCG19]. For instance, "the system must track and report sales" is a business requirement.

Technical Requirements: These detail the technical constraints and specifications [JH08], such as "the system must integrate with a specific database" or "the system must be developed using a specific programming language."

Types of Dependencies

According to researchers [HGC21], dependencies in software requirements illustrate the interconnections and prerequisites among various requirements. They include:

Functional Dependency: Where one requirement relies on the completion of another [CSL+01, AAN15, MAL+20]. For instance, the "checkout process" depends on "add product" and "cart management" functionalities.

Data Dependency: Where a requirement needs data generated or used by another [WLX21]. An example is the "order confirmation" relying on "order information."

Timing Dependency: Relating to the sequence of operations, such as "checkout" occurring after "add product" and "cart management" [WLX21].

Performance Dependency: Where the performance of one requirement impacts another [MSMS14, SHSM14]. For example, "high number of concurrent users" affects "system response time".

Security Dependency: Where the security of one requirement influences another, like "user authentication" impacting "data privacy" [SHSM14, JH08, EG04].

2.2 Related work

The evaluated articles together discuss the complex interdependencies within software requirements, proposing several techniques to effectively manage and optimize these interactions for improved software development. The articles suggest utilizing Semantic Web technologies, collaborative prioritizing strategies, and game theory to automatically detect and resolve conflicts and dependencies in requirements. The author highlighted the significance of well-defined Software Requirements Specification (SRS) papers and investigated methodologies for identifying conflicts and interdependencies among functional and non-functional requirements [BS19]. In addition, the use of ontology-based models and visualization tools is emphasized to enhance comprehension and facilitate communication of requirement relationships [SHSM14]. The impact of user interactions on system design is acknowledged in requirements engineering, with consideration given to social dependencies [MAL+20]. Collectively, these studies provide extensive approaches to enhance requirement management, hence assuring the effectiveness and efficiency of software development [1].

A software requirements specification (SRS) is a description of a software system to be developed. It is modelled after a business requirements specification. A software requirements specification sets out functional and non-functional requirements and may include a set of use cases that describe the user experience that the software must provide to the user for an ideal user experience [TSDL18].

The software document creates the basis for an agreement between customers and contractors or suppliers about how the software product should function (in a market-oriented project, these roles may be played by the marketing and development departments). A software requirements specification is a rigorous assessment of requirements before more specific system design steps, and its purpose is to reduce the number of subsequent iterations. It should also provide a realistic basis for estimating product costs, risks and schedules. When used correctly, software requirements specifications can help prevent software project failure.

A software requirements specification document contains a list of requirements sufficient for the development of a project. To derive requirements, the developer must have a clear and complete understanding of the products being developed [BS19]. This is achieved through detailed and continuous interaction with the project team and the customer throughout the software development process. The SRS is also important because it is the only source of information that prevents misunderstandings both between the project manager and the team and between the customer and the outsourcing company [TSDL18]. In complex projects, it is common to separate the system requirements specification from the software requirements. Although the terms "Software Requirements Specification" and "System Requirements Specification" are sometimes used interchangeably under the acronym SRS, the two concepts are not synonymous.

[1] https://docs.google.com/spreadsheets/d/15LmxNMbJjuvM-rvQxrq75KAHZS0zA40_PFD-DFRHWa8/edit?usp=sharing

3 Research methodology

The research methodology for this thesis is structured into three main phases, as illustrated in Figure 2. The process is iterative, with opportunities to refine the taxonomy based on the evaluation results before finalizing it. In what follows, the three key phases of the methodology are briefly discussed:

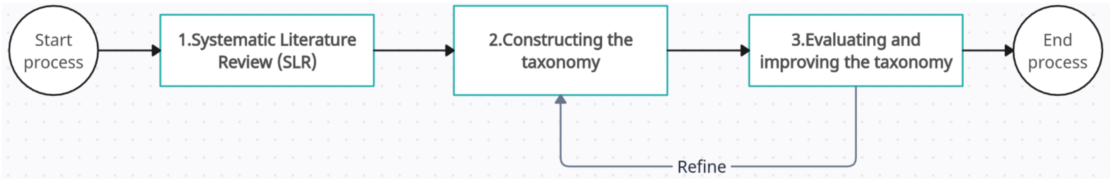

Figure 2. Research methodology.

3.1 Systematic Literature Review

This phase aimed at gathering data on existing requirements dependencies in relevant literature, existing taxonomies and the flaws of these taxonomies. Here, the direction of research questions is identified and research should be carried out using these questions. Based on the findings of the current study, there were 3 research questions identified and requirement dependencies and types were analyzed.

In line with the questions, criteria were established with the literature review conducted systematically.

3.2 Creating the Taxonomy

The second phase is to develop a new taxonomy in the application, based on the findings of the deficiencies and limitations of the existing literature review. The requirement dependencies related to taxonomy formation are found to be classified into the following main categories and then further subcategories. These categories, logical, time and other, are presented in the formation of the new taxonomy using the dependencies identified in the literature review section.

3.3 Evaluation of the New Taxonomy:

The proposed taxonomy for requirements interdependencies was evaluated against three other frameworks — iStar 2.0, Goal-Oriented Requirements Engineering (GORE), and Emotion-led Modeling. This comparison highlighted the taxonomy's strength in covering a broad range of interdependencies and its adaptability to established concepts.

Enhancements were made by incorporating the dependencies, improving its ability to assess the impact and sequence of requirements. This iterative refinement underscores the taxonomy's commitment to continuous improvement and its effectiveness in addressing the complexities of modern software systems.

4 The SLR to identify the most relevant studies to answer the RQs

To identify the most relevant and mature studies concerning requirements interdependencies, a Systematic Literature Review (SLR) was conducted. Following [Kit04, BS07a], the SLR process (shown in Figure 3) consists of three main phases: 1. Planning the review, in which I formulate the research questions (RQs), and I define the review protocol. 2. Conducting the review through a search process after identifying the search terms and the literature resources, followed by study selection activity, and 3. Reporting the results of the review by extracting detailed information from search results, and then using it to answer the research questions. In what follows, I describe each of these phases.

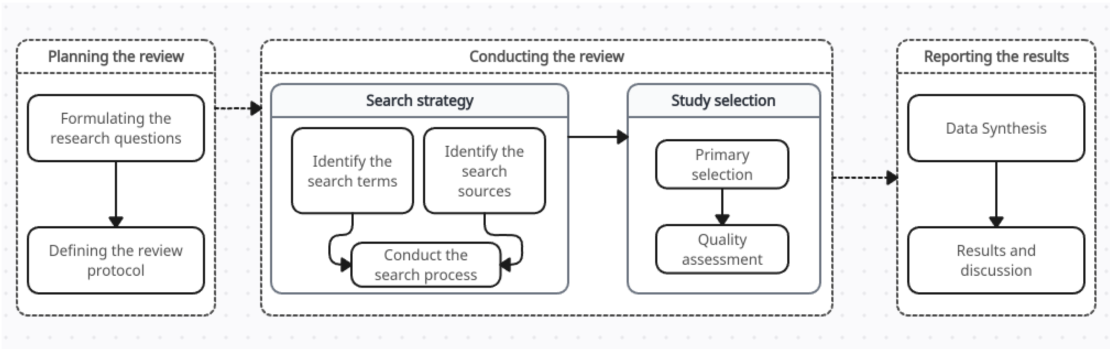

Figure 3. The systematic literature review process.

4.1 Planning the review

This phase is crucial for the review's success, as it is during this phase that I establish the research objectives and the review's methodology. This encompasses three primary activities:

Formulating the research questions is a crucial activity since these questions serve as the basis for developing the overall approach for a systematic review [Kit04]. The main aim of this review is to identify the most mature studies relevant to study. In this context, I used the first three research questions for the thesis mentioned earlier in section 1.3 Research Questions.

RQ1: What are the existing types of requirements interdependencies?

RQ2: How are dependencies between existing requirement types used across different requirement types?

RQ3: What are the coverage limitations of existing studies?

Defining the review protocol. A review protocol specifies the strategy that will be used to search for relevant studies; study inclusion and exclusion criteria, and study selection criteria. In what follows, I discuss how I performed each of these activities Table 1.

Table 1. Inclusion and exclusion criteria.

Exclusion criteria	**Inclusion criteria**
EC1: Papers that are not published in English	IC1: Papers related to at least one of the research questions
EC2: Papers that are not peer-reviewed. (i.e., not published in a conference or journal)	IC2: Only papers for which the full text is available will be included, ensuring that a thorough review can be conducted.
EC3: If a paper has several versions, only the most complete one is included.	IC3: Papers focusing on software/requirements engineering or systems engineering domains to maintain relevance to the field of study.

4.2 Conducting the review

This phase is composed of two main activities: 1 - search strategy; and 2 - study selection, where each of them is composed of several sub-activities.

4.2.1 Search strategy

This activity aims to find studies related to the research questions using an objective and repeatable search strategy. The search activity consists of three main sub-activities:

Identify the search terms. I derived the main search terms from the research questions. In particular, I used the Boolean AND to link the major terms, and I used the Boolean OR to incorporate alternative synonyms of such terms. The resulting search terms are:

("Requirements" OR "requirement") AND ("dependency" OR "dependencies") AND ("relation" OR "relations" OR "relationship" OR "relationships")

Identify the literature resources. I have selected several electronic database sources, namely IEEE Xplore, ACM Digital Library, Web of Science, and Scopus as they index the main scientific publications in the fields of software and requirements engineering.

Conduct the search. I have used the search terms to search the selected electronic database sources, and all returned studies were considered.

4.2.2 Study selection

Using the search terms to search the electronic database sources, returned 265 papers for Scopus, 411 papers for IEEE Xplore, 1052 for WoS and 59 papers for ACM Digital Library resulting in 1787 papers. After removing duplicated papers I had 1485 non-duplicated papers. I read the titles, and abstracts and skimmed through the rest of the sections of the papers, applied inclusion and exclusion criteria (shown in Table 1). This comprehensive approach ensured that the final selection of papers was highly relevant and contributed significantly to answering the research questions. The remaining 106 papers were fully read, and only papers that contained sufficient information to contribute to the answer to at least one of the quality assessment questions were included (shown in Table 2). This resulted in the selection of 30 papers [2]. 8 papers were excluded from the analysis because they did not address any relevant requirements dependencies/relationships. A simplified representation of the search and selection process is shown in Figure 4.

Table 2. Quality assessment questions.

Questions
Q1: Are the objectives of the proposed work related to eliciting, capturing, modelling, and/or analyzing requirements?
Q2: Are the proposed dependencies/relationships clearly defined?
Q3: Does the work propose sufficient concepts to capture dependencies/relationships among requirements?
Q4: Have the dependencies/relationships been applied to/within a project/-case study, or have they been justified by appropriate examples?

4.3 Reporting the results

The final phase of the systematic review involves summarizing the results, and it consists of two main activities:

[2] https://docs.google.com/spreadsheets/d/15LmxNMbJjuvM-rvQxrq75KAHZS0zA40_PFD-DFRHWa8/edit?usp=sharing

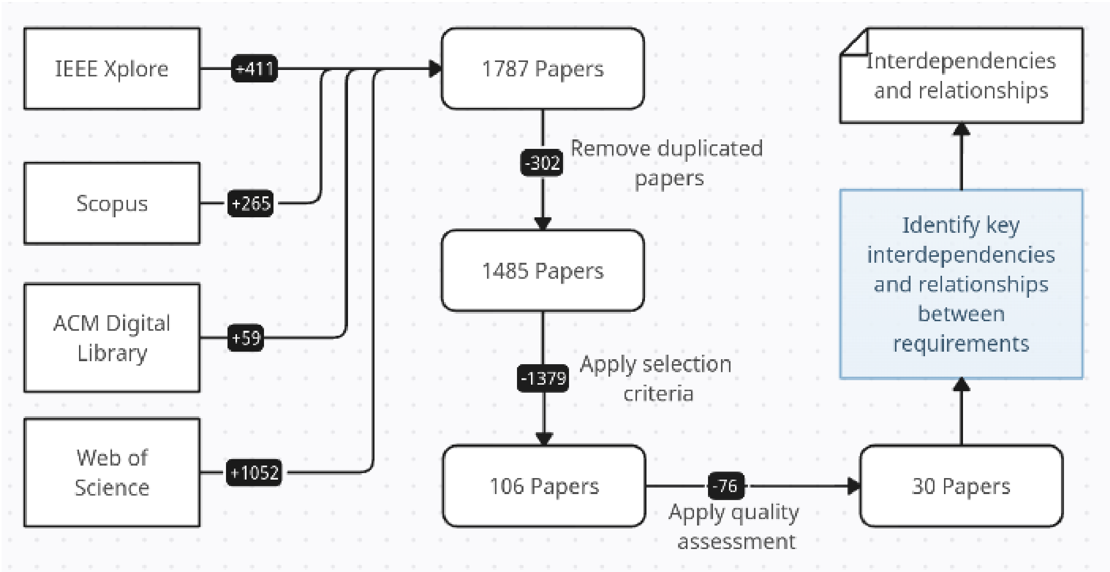

Figure 4. Paper search and selection process.

4.3.1 Data synthesis

This part aims at combining the findings of the selected studies in a way that allows answering the research questions. To facilitate this activity, each of the selected papers has been analyzed and its contribution to our RQ has been summarized. In what follows, I describe how data synthesis was executed: I extracted the data for answering RQ1 from the requirements dependencies that have been identified in the selected papers (shown in Table 3). Table 3 includes all identified relationships/dependencies in the selected papers, where relationships/dependencies are shown as **BOLD** representing key relationships/dependencies that have been selected based on their importance for capturing relationships/dependencies. To answer RQ2, I analyzed how each key identified requirements dependency has been used in the relevant literature. Table 4 includes key relationships/dependencies identified in the SLR accompanied by a brief description of each of them. Finally, I answered RQ3 by categorizing the selected studies into four groups based on their coverage of key dependency categories papers (shown in Table 5). I have categorised relationships/dependencies into 4 groups (the left two columns in Table 3). The identified studies do not completely cover all categories of the relationships/dependencies categories. For example, some studies (e.g., [CSL+01, MCG19]) completely cover FR and NFR relationships/dependencies, but partially cover "FR and NFR", and General relationships/dependencies, some (e.g., [KBV12, SJ05]) cover General relationships/dependencies, but partially cover FR and NFR relationships/dependencies, and totally do not cover "FR and NFR" relationship/dependencies (Table 5).

Requirements/Dependencies			[CSL+01]	[KBV12]	[AAN15]	[RBA10]	[SAC19]	[MSMS14]	[HGC21]	[WLX21]	[SJ05]	[IB01]	[GG19]	[MCG19]	[SHSM14]	[JH08]	[EG04]	[SK20]	[MAL+20]	[DFH16]	[vL01]	[MPLL+15]	[TSDL18]	[MVM00]
	FR	**AND**	✓		✓	✓		✓	✓	✓			✓	✓			✓	✓		✓	✓	✓		
		TEMPORAL	✓																					
		REPLACE								✓														
		OR	✓		✓	✓		✓	✓	✓			✓	✓			✓	✓		✓	✓	✓		
	NFR	**BEFORE**			✓																	✓		
		COMPLIANCE		✓														✓			✓			
		WISH																			✓			
		CONTRIBUTION					✓											✓		✓	✓			
	FR and NFR	**CONTINUANCE**		✓																				
		NON-INVERTABLE																						✓
		CONSEQUENTIAL		✓																				
		CONFLICT			✓						✓				✓		✓	✓			✓		✓	
		CONCURRENCY									✓													
		COOPERATION		✓													✓				✓			
		CVALUE	✓																					
		DEPENDENCE													✓								✓	
		ICOST	✓					✓				✓												
		SOCIAL																	✓					
		EXCLUSION								✓	✓													
	General	**REQUIRES**	✓					✓		✓										✓		✓		
		COORDINATE													✓								✓	
		CONTRADICTION													✓								✓	
		CONTRACTUAL		✓																				
		S-COST										✓												
		CAUSE-EFFECT									✓													
		TIME									✓													
		SUPPORT			✓					✓				✓							✓		✓	
		ABSTRACTION									✓													
		EQUIVALENCE								✓														
		SATISFIES								✓										✓	✓			

Table 3. Dependencies in the papers (**BOLD** represent key relationships/dependencies that have been selected)

Dependency	Description
AND	has been mainly used to refine top-level functional requirements into multiple sub-requirements [CSL+01, AAN15, RBA10, MSMS14, HGC21, WLX21].
OR	has been mainly used to decompose higher-level functional requirements into several lower-level requirements [CSL+01, AAN15, RBA10].
ICOST	refers to the impact of implementing one requirement on the cost of implementing another requirement, either increasing or decreasing the expense [CSL+01, MSMS14, IB01].
BEFORE	has been used to make sure that one requirement is implemented before another can be satisfied [AAN15].

Dependency	Description
CONTINUANCE	is a sequence of actions where one requirement triggers another, ensuring a logical and orderly development process [KBV12].
CONSEQUENTIAL	is the connection between two requirements where altering one requirement necessitates a corresponding change or fulfilment of the other [KBV12].
CONFLICT	arises when two or more requirements are incompatible and must be resolved to ensure a coherent and functional system [SHSM14, EG04, SK20, vL01].
COOPERATION	occurs when two or more requirements collaborate to achieve a shared goal, where the effective execution of one requirement enhances or enables the execution of another [KBV12, EG04, vL01].
CVALUE	has been used to impact one requirement affects the perceived value of another, influencing development prioritization and the overall value proposition of the system [CSL+01].
EXCLUSION	occurs when the implementation of one requirement is explicitly prohibited or invalidated by another, often due to mutual exclusivity or conflicting nature [SJ05].
REQUIRES	reqrepresents a relationship where the fulfillment of one requirement is contingent upon the satisfaction of another, indicating that one requirement relies on another to function effectively [CSL+01, MSMS14, WLX21, DFH16, MPLL+15].
CONTRACTUAL	refers to the linkages between requirements where one requirement necessitates the transmission of data to another, ensuring the accurate flow of information for both functional and non-functional aspects [KBV12].
SUPPORT	the relationship where the successful implementation of one requirement is essential for the effective achievement of another, applicable to both functional and non-functional needs [AAN15, vL01].
COMPLIANCE	is the connection between requirements that come from the necessity to conform to laws, rules, standards, or policies [KBV12, SK20, vL01].
SATISFIES	the relationship where the implementation of a requirement ensures the fulfillment of the specified objectives or criteria of another requirement [WLX21, DFH16, vL01].
CONTRIBUTION	accumulate evidence for qualities, determining whether they are fulfilled or denied [SAC19, DFH16, vL01].

Table 4. Dependencies in the taxonomy.

Table 5. A coverage of dependency categories in papers.

Paper	**FR**	**NFR**	**FR and NFR**	**General**
[CSL+01]	●	●	◒	◒
[KBV12]	◒	◒		●
[AAN15]	◒	◒	●	◒
[RBA10]		◒	◒	
[SAC19]		●	◒	
[MSMS14]		●		
[HGC21]	◒	◒		◒
[WLX21]	◒	◒		
[SJ05]	◒	◒		●
[IB01]				◒
[GG19]	◒	◒	◒	◒
[MCG19]	●	●	◒	◒
[SHSM14]	●	◒		
[JH08]		●		
[EG04]	◒	◒		◒
[SK20]		◒	◒	
[MAL+20]		◒	●	
[DFH16]	◒	◒		◒
[vL01]	◒	●		◒
[MPLL+15]	◒	◒		
[TSDL18]	◒	◒	◒	●
[MVM00]		◒		
● covered; ◒ partially covered; not covered				

4.3.2 Review results and discussion

The final results of SLR are placed in the table and are available at the link [3] below. In what follows, I present and discuss the findings of this review concerning each RQ in its corresponding step, as follows:

RQ1: *What are the existing types of requirements interdependence?* To answer this question, I have focused on identifying the most relevant and frequently cited require-

[3] https://docs.google.com/spreadsheets/d/15LmxNMbJjuvM-rvQxrq75KAHZS0zA40_PFD-DFRHWa8/edit?usp=sharing

ments and dependencies in the selected papers. Overall, I have found 30 requirements/dependencies after analyzing the 30 selected studies in SLR (shown in Table 4).

The identified dependencies cover diverse aspects of requirements interdependencies as they were derived from studies from different domains. For instance, some dependencies focus on capturing relationships between functional requirements in software development, other dependencies capture relationships between non-functional requirements.

Moreover, some identified dependencies involve an interaction between functional and non-functional requirements, where one requirement belongs to the functional category and the other to the non-functional category. This type of relation highlights the interdependency between what a system must do and the qualities or constraints under which the system must operate. The other type of dependency pertains exclusively to interactions where both sides of the relationship are either functional or non-functional requirements. In other words, this relation exists either between two functional requirements or between two non-functional requirements. This distinction emphasizes the homogeneity of the requirement types involved in the dependency, ensuring that the interaction is confined within the same category of requirements, whether functional or non-functional.

Also, there are some infrequently identified dependency types, such as Stakeholder-Based Dependencies, Inter-Actor Dependencies, and Resource-Based Dependencies. *Stakeholder-Based Dependencies* are based on different stakeholders' priorities and needs. These include conflicts between different stakeholders' requirements, where prioritizing one stakeholder's needs might negatively impact another's. *Inter-Actor Dependencies* are examined using game theory to analyze how different actors (stakeholders) with opposing non-functional requirements interact and how these interactions influence decision-making. *Resource-Based Dependencies* are based on the availability of resources, where certain requirements are dependent on the completion or availability of resources allocated to other requirements.

RQ2: *How are dependencies between existing requirement types used across different requirement types?* Dependencies between different requirement types—specifically functional and non-functional requirements—play a crucial role in ensuring that the overall system design and development are coherent, efficient, and aligned with both user needs and system constraints (depicted in Table 4). These dependencies are utilized in several key ways:

Intra-Type Dependencies: *Within Functional Requirements* are used to define sequences and hierarchies of operations. For example, before a system can generate a report (a functional requirement), it must first gather and process the necessary data, illustrating a dependency that governs the order of functionality implementation.

Within Non-Functional Requirements are employed to balance different quality attributes. For instance, enhancing "system security" might require adjustments in "system performance" due to the additional overhead of security protocols, thus creating a dependency where one non-functional requirement affects another.

Cross-Requirement Type Dependencies: *Functional to Non-Functional Dependencies* are often leveraged to ensure that functional requirements (what the system should do) are fulfilled in a way that meets non-functional requirements (how the system should perform). For instance, a functional requirement such as "process payments" may be dependent on non-functional requirements like "ensure transaction security" and "maintain system performance under load." This ensures that the functional requirement is not only implemented but also operates under the desired quality conditions.

Non-Functional to Functional Dependencies can also dictate the design and implementation of functional requirements. For example, a non-functional requirement for "high system availability" might influence the functional requirements related to data redundancy and failover mechanisms, ensuring that the system remains operational under various conditions.

Conflict Resolution and Trade-offs: *Managing Conflicts* are crucial for identifying and resolving conflicts between different requirement types. For example, a non-functional requirement for "maximum usability" might conflict with stringent "security" requirements, as more security can sometimes reduce usability. Understanding these dependencies allows project teams to negotiate and prioritize requirements effectively.

Balancing Trade-offs help in balancing trade-offs between different requirement types. For example, improving system performance (non-functional) might require simplifying certain functions (functional), thus reducing complexity but potentially limiting functionality.

Facilitating Communication and Collaboration: *Stakeholder Alignment*. Understanding dependencies across requirement types helps in aligning stakeholder expectations. For instance, recognizing how a functional requirement supports or is constrained by non-functional requirements allows stakeholders to have a clearer understanding of trade-offs and compromises, facilitating better collaboration and decision-making.

RQ3: *What are the coverage limitations of existing studies?* Answering this question, I categorized the limitations of existing studies. Despite significant advancements in understanding and managing requirement dependencies, existing studies exhibit certain coverage limitations that impact the comprehensiveness and applicability of their findings.

Limited Scope on Functional and Non-Functional Requirements. As seen in Table 5, many studies predominantly do not focus on functional and non-functional requirements together. This results in a lack of comprehensive frameworks that fully integrate dependencies between functional and non-functional requirements, such as cost of security, usability, and performance, into the analysis of requirement dependencies. The absence of such integration limits the applicability of these studies in real-world scenarios where functional and non-functional attributes are crucial for system success.

Inadequate Consideration of Dynamic and Evolving Requirements. Existing research often assumes that requirements are static and unchanging, which is not reflective of real-world software development where requirements frequently evolve. This limitation restricts the utility of these studies in agile or iterative development environments where requirements are expected to change over time, necessitating a more dynamic approach to managing dependencies.

Lack of Comprehensive Tools and Techniques for Visualizing Dependencies. Although some studies propose tools and techniques for visualizing requirement dependencies, these are often fragmented and lack standardization. The absence of comprehensive, standardized tools limits the ability of practitioners to effectively communicate and manage requirement dependencies across diverse teams and stakeholders.

Inconsistent Methodological Approaches. There is a noticeable inconsistency in the methodologies used across studies to analyze requirement dependencies. This inconsistency makes it difficult to compare results, draw general conclusions, or develop universally applicable frameworks, thereby limiting the overall impact and utility of the research.

5 Constructing the Taxonomy for Requirements Interdependencies

As previously discussed, the suggested novel taxonomy classifies requirement relationships or dependencies in a project into four primary categories (shown in Figure 5). Functional requirement relationships/dependencies (I consider relationship and dependency as synonyms in this thesis) refer to the associations between two functional requirements, which specify the objectives of the system. Non-functional requirement dependencies refer to the relationships between non-functional requirements that determine the execution of certain functions in the system, such as performance, security, or usability. The relationships and dependencies between functional and non-functional requirements encompass the relationships between a functional requirement and a non-functional need. These relationships ensure that the system's functionality is in line with performance or quality standards. Finally, some general relationships and dependencies apply to both functional and non-functional requirements, ensuring that all forms of requirements are evaluated in a unified way. Gaining a comprehensive understanding of these relationships is essential for the effective handling of requirements and the achievement of successful project outcomes.

5.1 Functional requirement dependency

Functional dependencies pertain to the interdependencies among several functional requirements of a software system, wherein the execution or functioning of one requirement is reliant on the execution or functioning of another. This idea is crucial in software development as it guarantees that capabilities are built logically and coherently, hence avoiding integration difficulties and assuring the proper functioning of dependent features [AAN15, HGC21, SJ05, IB01].

***And* dependency** is when a top-level requirement is broken down into multiple sub-requirements, ensuring that the implementation of the primary requirement depends on the concurrent fulfilment of these sub-requirements [CSL$^+$01, AAN15, RBA10, MSMS14, HGC21, WLX21, GG19, MCG19, EG04, SK20, DFH16, vL01, MPLL$^+$15]. This method ensures that the top-level requirement's full functionality is realized through coordinated and cooperative implementation of its component pieces. *And* dependencies are particularly useful when complex requirements require intricate breakdowns into implementable parts. For instance, in the "The user must log in and enter their password correctly" requirement, the requirements sign in and password verification are in an *And* relation. If both are not met the user can not log in.

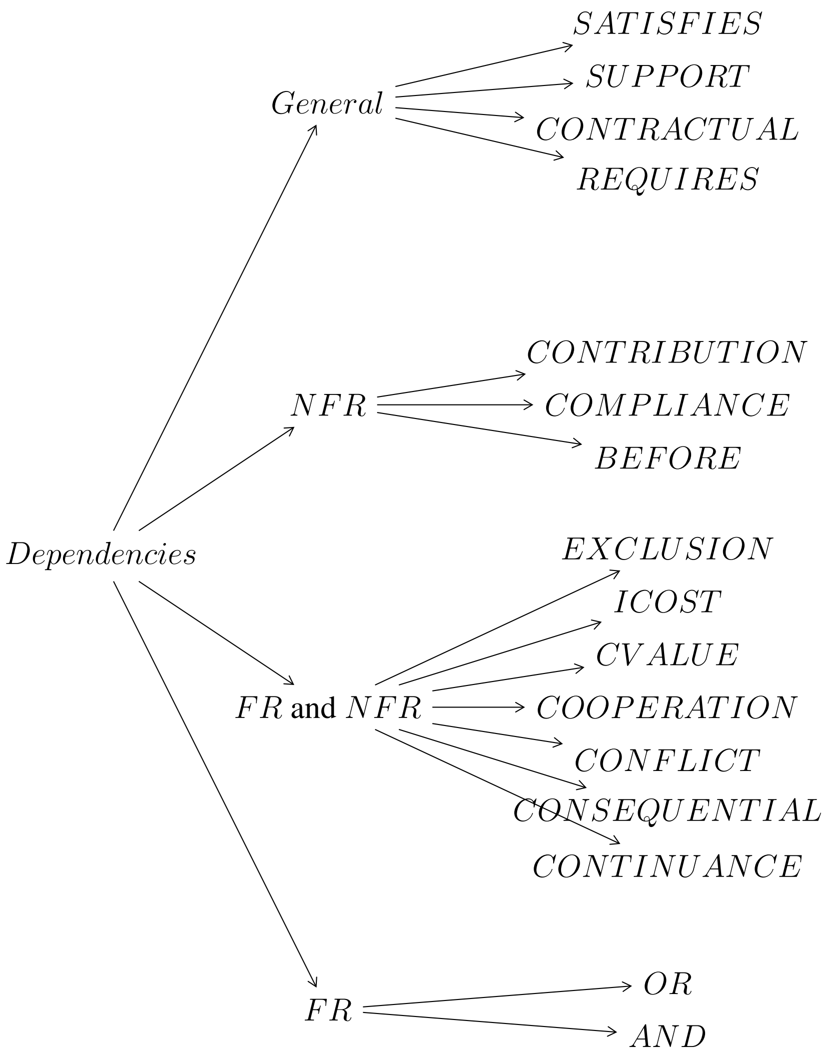

Figure 5. A proposed novel taxonomy.

***Or* dependency** When a higher-level requirement is decomposed into several lower-level requirements, *Or* dependencies are a means to create links between these requirements [CSL+01, AAN15, RBA10, MSMS14, HGC21, WLX21, GG19, MCG19, EG04, SK20, DFH16, vL01, MPLL+15]. The higher-level requirement may be satisfied by satisfying any one of these lower-level requirements. Different methods or techniques may be used to accomplish a certain target using this form of dependence, which provides flexibility in the implementation process. For instance for the requirement like "User can pay by credit card or pay with PayPal", in this case, requirements like "credit card payment" and "PayPal payment" have OR relationship. The user just has to be able to purchase something with any of these payment methods.

5.2 Non-functional requirement dependency

Non-functional dependencies refer to the interconnected links between non-functional requirements (NFRs) and the quality attributes of a software system [SAC19, MSMS14, HGC21, SHSM14, JH08]. These dependencies illustrate the interconnectedness of non-functional aspects such as scalability, usability, security, and performance. Enhancing one non-functional requirement (NFR) may potentially undermine another, underscoring the need to comprehend and control these interdependencies to maximize the overall quality of the software system.

***Before* dependency** is crucial in the requirements phase before software development, ensuring that one requirement (R1) is implemented before another (R2) can be satisfied [AAN15]. This relationship ensures that the dependent requirement (R2) has the necessary prerequisites for proper functioning. For instance, one of the steps of the registration is the read and approve of the Privacy Policy. In a requirement such as "User must first accept the Privacy Policy before signing up" the requirements 'Accept Privacy Policy' and 'sign up' have a *Before* relation. The Privacy Policy is to be accepted as a prerequisite for registration.

***Compliance* dependency** is the connection between requirements that come from the necessity to conform to laws, rules, standards, or policies [KBV12, SK20, vL01]. These dependencies guarantee that specific requirements are implemented by external guidelines or internal policies to ensure adherence to legal, regulatory, or standard requirements. A compliance dependency is when an entity's adherence to certain regulations or standards is contingent upon another factor or condition. R1 refers to compliance.R2 is considered to exist to two requirements, R1 and R2, when the fulfilment or compliance with R1 requires following a law, policy, or standard that affects R2. Non-Functional Requirements encompass performance, security, and usability that must conform to regulatory standards. As an illustration, when a program for a particular task needs to adhere to the required set of practices of the PCI DSS standard.

***Contribution* dependency** illustrates how requirements impact qualities, playing a crucial role in aiding analysts in choosing between various goals or tasks [SAC19, DFH16, vL01]. These dependencies accumulate evidence for qualities, determining whether they are fulfilled or denied. A quality is considered fulfilled if there is sufficient positive evidence and denied if there is significant negative evidence. Contributions are defined as relationships from a source requirement to a target quality, with the following types:

- Make: The source provides strong positive evidence for the satisfaction of the target.
- Help: The source offers weak positive evidence for the satisfaction of the target.
- Hurt: The source provides weak evidence against the satisfaction or supports the denial of the target.
- Break: The source offers strong evidence against the satisfaction or supports the denial of the target.

5.3 Functional and non-functional requirement dependency

***Icost* dependency** are the effects of one requirement's implementation on the cost of another's implementation [CSL+01, MSMS14, IB01]. More precisely, requirement R1 *Icost* requirement R2 states that putting R1 into practice will either make it more expensive to accomplish R2 or less expensive. These dependencies are essential to grasping the economic consequences of requirement interdependencies and to managing budgets efficiently during the software development process. Understanding how obtaining particular quality characteristics, such as security or performance, may affect the cost of accomplishing other non-functional goals is made possible by *Icost* dependencies for non-functional requirements.

***Continuance* dependency** is a sequence of actions where one requirement triggers another, ensuring a logical and orderly development process [KBV12]. It can be applied to both functional and non-functional requirements, dictating the order of tasks and conditions for successful project completion. For instance, if a person is logged in, they are considered as logged in. In a requirement such as "As long as the user is logged in, they are automatically logged in", the "login" and "autologin" requirements have a CONTINUANCE dependency.

***Consequential* dependency** is the connection between two requirements, R1 and R2, where changing R1 requires a corresponding change or fulfillment of R2 [KBV12]. For example, the implementation of a new function such as "User Registration" may need an update to the existing "Login" functionality. Similarly, enhancements in security protocols may entail adjustments to performance requirements in order to maintain the system's performance unchanged.

***Conflict* dependency** arises when there are incompatible two or more requirements, which need to be settled in order to provide a coherent and functional system [SHSM14, EG04, SK20, vL01]. Conflicts may result from functional requirements if some features have mutually exclusive implementation requirements or cannot be used concurrently. Dealing with non-functional requirements usually results in conflicts between multiple quality elements, including security against performance or usability against dependability. For instance, where there is a requirement like “User cannot log in to two different applications at the same time. ” In a requirement such as “user cannot log in to two different applications at the same time”, if the login to the first application is true then the login to the second application is false, in other words, they have a conflict dependency. The user cannot key in his username and password and use both applications simultaneously.

***Cooperation* dependency** arises when two or more requirements collaborate to accomplish a shared goal [KBV12, EG04, vL01]. This form of dependence signifies that the effective execution of one requirement improves or enables the execution of another, hence contributing to the overall objectives of the system. Cooperation dependencies are defined as a collaborative connection in which the requirements of each party mutually reinforce each other to achieve a combined output that would be less efficient if the requirements were independent. Functional requirements entail the collaboration of several system components to execute intricate operations. In the context of non-functional requirements, cooperation refers to the collaboration of several quality criteria, such as performance and reliability, to improve the system's resilience. For example, where a requirement like the following is defined: ‘Two different systems must complete a transaction by sharing user data.’ Then, in a requirement like that where the ’first system performs transaction’ and ’second system shares data’, the two have a *Cooperation* dependency. Each system relies on the other to be functional; therefore, if the two systems are not in harmony, the transaction cannot work out.

***Cvalue* dependency** refers to a specific form of requirement that impacts the value of another requirement [CSL$^+$01]. The interdependence between various factors can influence the prioritization of development and the comprehension of how distinct system attributes contribute to its overall value proposition. *Cvalue* dependencies encompass both functional and non-functional requirements, exerting an influence on the overall effectiveness and user satisfaction of a system. Improvements in a specific aspect of quality, such as performance, might enhance the value or accessibility of another, thus impacting the total value proposition of the system. For example, in the requirements like “Customizable options available in an application can increase a user's perception of value. “ the “customization options” and “user value” requirements have *Cvalue* dependency. This means that customization options decided by the maker are going to influence how the user perceives the value of the product that is being sold to him.

***Exclusion* dependency** is a dependency in which the implementation of one requirement is explicitly prohibited or invalidated by another [SJ05]. This is essential in cases where specific functionalities are unable to coexist within a system as a result of mutual exclusivity or conflicting nature. For example, a requirement that specifies "The system must support single-factor authentication" may be one of the *Exclusive* requirements. "The system must support multi-factor authentication" if the system is intended to operate with only one form of authentication due to cost or complexity requirements. In the same vein, the requirement that "The system must be optimized for maximum speed" may be *Exclusive* of the requirement that "The system must provide detailed logging and monitoring" because detailed logging could introduce performance overhead that conflicts with the speed optimization objective.

5.4 General requirement dependency

***Requires* dependency** represents a dependency between two requirements, where the satisfaction of one requirement relies on another requirement [CSL$^+$01, MSMS14, WLX21, DFH16, MPLL$^+$15]. This kind of dependence indicates that a specific need R1 requires the implementation of another requirement R2 in order to function effectively. In essence, the fulfilment of R1 is contingent upon the satisfaction of R2, suggesting a hierarchical or interdependent connection where R2 provides support for R1. Dependencies can be applied to both functional and non-functional needs. *And/Or* dependency is a special type of the *Requires* dependency. In a requirement like ‘the user has to sign in to upload a file’ the ‘sign in’ and ‘file upload’ requirements are dependent on one another in the sense that the latter cannot be satisfied until the former has been met. Before uploading files the user is required to log in.

***Contractual* dependency** refers to the linkages between requirements, where one requirement necessitates the transmission of data to another requirement, hence assuring the accurate flow of information [KBV12]. They can be used for both functional and non-functional aspects, such as payment processing and user authentication. Within a functional framework, user authentication might require payment processing data, whereas non-functional requirements may need data from security measures.

***Support* dependency** defines the relationship where the successful implementation of one requirement is essential for the effective achievement of another [AAN15, vL01]. This concept involves the process of transforming high-level requirements into specific sub-requirements, intending to establish key components before working on interconnected components. Support dependencies can be applied to both functional and non-functional needs. For instance, functional requirements called "Database Setup" must be able to fulfil a requirement called "Data Retrieval". Similarly, a non-functional requirement called "System Security Policy" must be established to enable measures for "Data Access Control".

***Satisfies* dependency** in requirements engineering is an essential component that guarantees the satisfaction of both functional and non-functional requirements [WLX21, DFH16, vL01]. A dependence is a relationship that arises when the implementation of a requirement guarantees that the specified objectives or criteria of the requirement are fulfilled. This reliance is crucial for attaining the objectives outlined by the requirement. For instance, a need specifying "The system must allow users to log in" might be fulfilled by guaranteeing that only authorized users can access the system, while a requirement mandating the encryption of user data can be fulfilled by assuring the confidentiality of the data.

6 Evaluation

This section provides a detailed comparative analysis of the proposed taxonomy for requirements interdependencies against three other taxonomies in Table 6: iStar 2.0 [DFH16], Goal-Oriented Requirements Engineering (GORE) [vL01], and Emotion-led Modeling for People-Oriented Requirements Engineering [MPLL$^+$15]. The first two have been chosen because they are established and influential works in the RE domain, and provide a comprehensive set of relationships/dependencies among requirements. The third work has been selected because it is a comparatively new work and deals with a different set of requirements.

The strength of the proposed taxonomy lies not only in its initial comprehensiveness set of relationships/dependencies but also in its demonstrated adaptability and responsiveness to insights gleaned from established frameworks. The comparative analysis underscored the taxonomy's ability to encompass a wide spectrum of interdependencies, aligning seamlessly with core concepts from iStar 2.0, GORE, and Emotion-led Modeling. This alignment serves as a testament to the taxonomy's robust foundation and its potential for integration into diverse requirements engineering practices.

However, the evaluation process also unearthed opportunities for further enhancement of my proposed taxonomy. Recognizing the value of incorporating diverse perspectives, the taxonomy was expanded to include the *Before* dependency. Though initially absent from the proposed taxonomy, this dependency was identified as crucial in the evaluated studies, highlighting the importance of remaining receptive to insights from existing frameworks. The integration of *Before* dependency enhances the taxonomy's ability to model temporal sequencing and prerequisite relationships between requirements, fostering a logical and well-ordered implementation strategy.

Dependency	**Description in proposed taxonomy**	**i-star [DFH16]**	**GORE [vL01]**	**Emotion-led [MPLL$^+$15]**
AND (FR)	is when a top-level requirement is broken down into multiple sub-requirements, ensuring that the implementation of the primary requirement depends on the concurrent fulfilment of these sub-requirements.	AND refinement	AND refinement	Goal Decomposition(AND dependencies)

Dependency	Description in proposed taxonomy	i-star [DFH16]	GORE [vL01]	Emotion-led [MPLL+15]
OR (FR)	When a higher-level requirement is decomposed into several lower-level requirements, OR dependencies are a means to create links between these requirements.	Inclusive OR refinement	OR-refinement	
BEFORE (NFR)	is crucial in the requirements phase before software development, ensuring that one requirement (R1) is implemented before another (R2) can be satisfied.		Depends On	
COMPLIANCE (NFR)	is the connection between requirements that come from the necessity to conform to laws, rules, standards, or policies.		Security	
ICOST dependency (FR and NFR)	are the effects of one requirement's implementation on the cost of another's implementation.			
CONTINUANCE (FR and NFR)	is a sequence of actions where one requirement triggers another, ensuring a logical and orderly development process.		Operationalization	
CONSEQUENTIAL (FR and NFR)	is the connection between two requirements, R1 and R2, where changing R1 requires a corresponding change or fulfilment of R2.		Responsibility	
CONFLICT (FR and NFR)	arises when there are incompatible two or more requirements, which need to be settled to provide a coherent and functional system.		Conflict Resolution	
COOPERATION (FR and NFR)	arises when two or more requirements collaborate to accomplish a shared goal.			

Dependency	Description in proposed taxonomy	i-star [DFH16]	GORE [vL01]	Emotion-led [MPLL+15]
CVALUE (FR and NFR)	refers to a specific form of requirement connection in which the customer value of one need is modified by another.			
EXCLUSION (FR and NFR)	is a dependency in which the implementation of one requirement is explicitly prohibited or invalidated by another.		Inhibits	
REQUIRES (GR)	represents a dependency between two requirements, where the satisfaction of one requirement relies on another requirement.	NeededBy	Requires	
CONTRACTUAL (GR)	refers to the linkages between requirements, where one requirement necessitates the transmission of data to another requirement, hence assuring the accurate flow of information.			
SUPPORT (GR)	defines the relationship where the successful implementation of one requirement is essential for the effective achievement of another.		Support	
SATISFIES (GR)	in requirement engineering is an essential component that guarantees the satisfaction of both functional and non-functional requirements.	Qualification NFR (relates quality to its subject)	Satisficing	

Dependency	Description in proposed taxonomy	i-star [DFH16]	GORE [vL01]	Emotion-led [MPLL+15]
CONTRIBUTION (NFR)	to capture the positive and negative contribution of requirements to goals.	Contribution (represents the effects of intentional elements on qualities)	Contribution	Goal Contribution(Contribute Dependencies)
				Implicit Dependencies

Table 6. Evaluation of suggested taxonomy.

This iterative refinement process exemplifies the dynamic and evolutionary nature of the taxonomy. It underscores the importance of continuous improvement and adaptation in the field of requirements engineering. By actively seeking and integrating valuable concepts from established frameworks, the taxonomy maintains its relevance and applicability amidst the ever-evolving landscape of software development. The deliberate inclusion of *Contribution* and *Before* dependencies serves as a testament to the taxonomy's commitment to growth and its capacity to address the nuanced complexities of modern software systems.

Furthermore, this refinement process reinforces the notion that taxonomies, much like the systems they model, should be flexible and open to enhancement. By embracing new perspectives and incorporating valuable insights, the taxonomy ensures its continued effectiveness in navigating the intricate web of requirement interdependencies, ultimately contributing to the development of high-quality software systems that meet the diverse needs of stakeholders.

Suggested taxonomy covers a wide range of interdependencies, including *And*, *Or*, *Before*, *Compliance*, *Icost*, *Continuance*, *Consequential*, *Conflict*, *Cooperation*, *Cvalue*, *Exclusion*, *Requires*, *Contractual*, *Support*, *Satisfies*, and *Contribution*. This comprehensive set of interdependencies ensures that all necessary relationships between requirements are captured and managed. By aligning with principles from iStar 2.0, GORE and Emotion-led Modeling, the proposed taxonomy ensures that each requirement is strategically aligned with system goals and managed in a structured manner.

7 Validation

This section validates the proposed taxonomy by demonstrating its ability to encompass a broader range of relationships/dependencies than those supported by the existing MGM framework. The validation process involves two key steps: first, highlighting the relationships and dependencies that are already supported by the existing framework, and second, illustrating how the proposed taxonomy introduces additional relationships that extend the coverage of the requirements model.

By incorporating these additional relationships/dependencies, the proposed taxonomy offers a more comprehensive approach to modelling the requirements, thereby enhancing the robustness and expressiveness of the requirements model. This expansion allows for a more detailed and nuanced understanding of the interconnections between functional, quality, and emotional goals, ultimately leading to a more exhaustive and well-rounded representation of the system's requirements.

The following subsections detail the relationships supported by the existing MGM framework and introduce new relationships that are unique to the proposed taxonomy. This comparison underscores the added value of the proposed taxonomy in capturing a wider array of interactions/dependencies within the model.

The example is abstracted from the PHArA-ON (Pilots for Healthy and Active Ageing in Europe) H2020 European project[4], The main aim of the PHArA-ON project is to actualize smart and active living for Europe's ageing population and make such smart and active living a reality.

Figure 6 depicts the requirements model for "support well-being of older adults" represented in the Motivational Goal Model (MGM) language [MPLL^{+}15]. The model is developed starting with specifying the top-level functional goal, namely "support the well-being of older adults", which represents the main objective/goal the system needs to achieve. Then, the top-level goal is refined into three sub-goals: "Improve Digital Skills", "Participate in the community", and "Provide cognitive stimulation". These sub-goals are characterized by several quality and emotional goals, i.e., these quality and emotional goals are attached to relevant functional goals. Finally, the roles responsible for achieving goals are added to the model.

As seen in the diagram and discussed in the previous section, the MGM is not that descriptive for capturing relationships/dependencies among requirements.

In what follows, I apply the proposed taxonomy to the same example showing how the new taxonomy can tackle the shortcomings in the motivational goal model concerning capturing the relationships/dependencies among requirements.

1 - Dependencies Supported by Both the Existing Framework and Proposed Taxonomy represent the foundational relationships and interactions that are acknowledged by

[4] https://www.pharaon.eu/

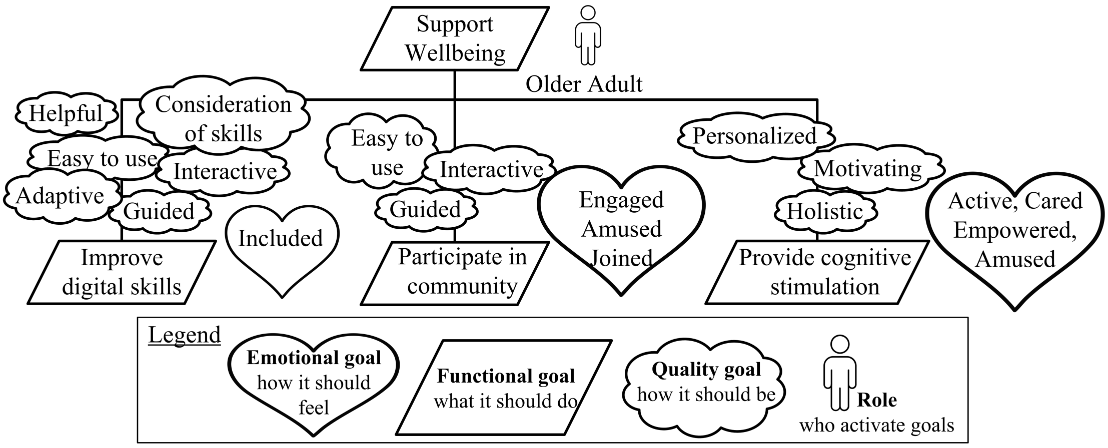

Figure 6. The motivational goal model for supporting the well-being of older adults [MMF+ay]

both the existing framework and the proposed taxonomy. These dependencies are critical in ensuring that the essential connections between functional, quality, and emotional goals are captured accurately. By supporting these shared dependencies, the proposed taxonomy aligns with established modelling practices, reinforcing the core structure of the requirements model. The following analysis demonstrates how these common dependencies contribute to a coherent and effective requirements model.

Contribute interdependency. Both the MGM framework and the proposed taxonomy underscore the importance of understanding how individual requirements or goals contribute to overarching objectives. While the MGM framework visually represents these contributions, the taxonomy introduces a more detailed classification system—comprising the categories of Make, Help, Hurt, and Break—to indicate the strength and nature of each contribution. For instance, the availability of the goal "Participate in the community" can be interpreted as making a significant positive contribution to the emotional goal of feeling "Joined."

Sequence (ordered) interdependency. The temporal ordering of requirements, which is implicitly represented in the MGM through the arrangement of goals, is explicitly addressed in the proposed taxonomy through the Before dependency. For example, in MGM (Figure 6) the functional goal of "Participate in the community" may necessitate the prior fulfilment of the goal "Improve Digital Skills." This sequential relationship ensures that older adults possess the requisite technological proficiency before engaging in online communities.

Support interdependency. Both frameworks recognize that certain requirements or goals facilitate the achievement of others. The MGM visually connects supporting elements, while the proposed taxonomy formalizes this relationship through the Support dependency. For instance, the quality goal "Adaptive" can support the functional goal "Provide cognitive stimulation" by tailoring the level of challenge to an individual's cognitive abilities, thereby enhancing the effectiveness of the intervention.

And interdependency is not explicitly visualized in the MGM, it is inherent in the model's structure, implying that multiple sub-goals or requirements may need to be fulfilled concurrently to achieve a higher-level goal. For example, to successfully "Participate in the community," an older adult may need to both "Improve Digital Skills" and overcome "Anxiety about technology" (an emotional goal). This aligns with the And dependency in my taxonomy, which emphasizes the necessity of fulfilling multiple requirements simultaneously.

2 - Dependencies Unique to Proposed Taxonomy introduces additional relationships not supported by the existing MGM framework, thereby expanding the scope and depth of the requirements model. These unique dependencies allow for a more detailed and nuanced representation of the interactions between different types of goals, addressing gaps and limitations within the existing framework. By incorporating these novel dependencies, the proposed taxonomy enhances the model's expressiveness and completeness, providing a more robust foundation for requirements engineering. The following discussion explores these unique dependencies and their implications for the requirements model.

Icost interdependency. While the MGM focuses primarily on functional, quality, and emotional aspects of goals, it does not explicitly address their economic implications. The proposed taxonomy addresses this gap with the Icost dependency, which highlights the potential cost impact of fulfilling certain requirements. For example, incorporating advanced accessibility features to accommodate a broader range of older adults (a quality goal) may increase development and maintenance costs, necessitating careful consideration during project planning.

Cooperation interdependency in the proposed taxonomy captures the collaborative nature of certain requirements, a nuance not explicitly represented in the MGM. This dependency underscores the necessity for collaborative efforts in achieving specific objectives, where multiple requirements work synergistically to meet the overall goal. In MGM (Figure 6), the functional goal of "Improve Digital Skills" may cooperate with the functional goal of "Participate in the community," as enhanced digital literacy empowers older adults to engage more actively in online communities.

Requires interdependency in the proposed taxonomy offers a more stringent interpretation of supportive relationships not implied in the MGM. It underscores that certain requirements are absolute prerequisites for others. For instance, the goal "Improve digital skills" (a functional goal) requires satisfying the "Helpful" (a quality goal) to represent a more robust model. This denotes a conditional relationship, wherein the achievement of one goal is contingent upon the fulfilment of another. Another example is the quality goal of "Secure data storage," which is a fundamental requirement before any personal health information can be collected or processed, ensuring user privacy and trust.

Or interdependency is also not supported by the MGM framework. It suggests flexibility in achieving goals through alternative pathways, which aligns with the Or dependency in the proposed taxonomy. For example, the emotional goal of "Included" might be achieved through various means, such as online social interactions or in-person community events, thereby offering older adults the option to engage in ways that align with their preferences and capabilities.

Moreover, Cvalue interdependency emphasizes the influence of certain requirements on the overall value proposition of the technology, an aspect not directly addressed in the MGM. This relationship demonstrates the impact of customer value adjustments on the interrelated requirements in the proposed taxonomy. For instance, providing personalized health monitoring and feedback (a quality goal) could significantly enhance the perceived value of the technology for older adults, thereby increasing its adoption and long-term use.

In conclusion, the application of the proposed taxonomy to the aforementioned example demonstrates the applicability and usefulness of the proposed taxonomy in producing more expressive and accurate requirements models. More specifically, by incorporating nuanced dependencies such as Icost, Cooperation, Cvalue, and Requires, alongside the foundational principles of the Motivational Goal Model (MGM), I were able to produce more accurate models, which provides a robust basis for informed decision-making. It facilitates the optimization of resources and supports the development of technology solutions that are not only effective and user-centered but also economically sustainable and attuned to the evolving needs and values of the target population.

8 Threats to validity

This section addresses potential threats to the validity of the study. Following the framework suggested by Runeson et al. [PM09], I categorize these threats into four types: construct, internal, external, and reliability.

1 - Construct Validity pertains to how well the study measures what it is intended to measure [PM09]. The importance of construct validity cannot be overstated, as it can directly impact internal validity [B.03]. One potential threat I identified is *Systematic Error*. This error could arise during the design and execution of the review. To mitigate this risk, I meticulously designed the review protocol based on established methodologies and adhered to it strictly throughout the review process.

2 - Internal Validity concerns the influence of unconsidered factors that might affect the outcomes of the study [PM09]. *Potential Publication Bias*: This common threat to systematic reviews occurs when studies with positive outcomes are more likely to be published than those with negative results [BS07b]. My review aimed to identify privacy-related concepts and relationships by examining relevant literature, which included both positive and negative findings. To counteract this bias, I established clear inclusion and exclusion criteria, along with stringent quality assessment measures during the selection of studies.

3 - External Validity. This aspect relates to the generalizability of the study's results [PM09]. *Completeness*: Achieving complete coverage of all relevant studies is nearly impossible. Despite this challenge, my review protocol and search strategy were carefully designed to encompass as many relevant studies as possible. Additionally, I may have unintentionally excluded some pertinent studies published in languages other than English. To address this, I manually reviewed the references of all primary selected studies to identify any that may have been overlooked in the initial search.

4 - Reliability concerns the consistency of the study's results if repeated by different researchers. The study's methodology, including search terms, sources, inclusion and exclusion criteria, and quality assessment questions, is fully documented[5]. This transparency ensures that other researchers can replicate the review and expect to obtain similar results.

[5] https://docs.google.com/spreadsheets/d/15LmxNMbJjuvM-rvQxrq75KAHZS0zA40_PFD-DFRHWa8/edit?usp=sharing

9 Possible further developments/ Limitations and future work

The suggested novel taxonomy for requirements interdependencies has great potential for further development but also faces some limitations that could be addressed through future work.

One potential development is to enhance traceability mechanisms within the taxonomy. By linking requirements more comprehensively to the design, implementation, and testing phases, the taxonomy could ensure that dependencies are maintained and managed effectively throughout the project lifecycle. Developing automated tools to track these relationships and provide real-time updates would significantly improve the ability to monitor progress and changes in requirements.

Another area for development is the integration of the taxonomy with agile methodologies. The current taxonomy can be adapted to be more compatible with iterative and incremental approaches commonly used in agile projects. This would involve refining the taxonomy to allow for continuous feedback and adjustment of requirements, thus making it more flexible and responsive to changes.

Despite these potential developments, the taxonomy also has some limitations that need to be addressed. One limitation is the complexity involved in managing and visualizing large numbers of interdependencies in big projects. Without adequate tool support, the manual effort required can be overwhelming, and there is a risk of missing critical dependencies.

Furthermore, while the taxonomy aims to cover various types of interdependencies, it might not fully capture the dynamic nature of requirements in rapidly changing environments. Future work could focus on enhancing the adaptability of the taxonomy to cater to evolving requirements and contexts.

In conclusion, while the taxonomy for requirements interdependencies provides a robust framework for managing complex relationships among requirements, there is significant potential for further development. Enhancing traceability, integrating with agile methodologies and developing automation tools are promising directions for future research and development. Addressing these areas could greatly enhance the effectiveness and applicability of the taxonomy, making it a more powerful tool for managing requirements in complex projects.

10 Conclusions

The thesis has successfully achieved its primary objective of developing a novel taxonomy for requirements interdependencies. The taxonomy, constructed through a systematic literature review and evaluation, offers a comprehensive and structured framework for understanding and managing the intricate relationships among requirements in software development projects. The 16 distinct dependency types, categorized into functional, non-functional, functional and non-functional, and general dependencies, provide a nuanced understanding of how requirements interact and influence each other. The taxonomy's effectiveness has been demonstrated through its alignment with established frameworks like iStar 2.0, GORE, and Emotion-led Modeling, showcasing its ability to capture and manage diverse interdependencies in various requirements engineering contexts. The taxonomy's emphasis on strategic prioritization and conflict resolution further enhances its practical value, enabling stakeholders to make informed decisions and ensure the successful delivery of software projects.

References

[AAN15] Ruba Alzyoudi, Khaled Almakadmeh, and Hutaf Natoureah. A probability algorithm for requirement selection in component-based software development. IPAC '15, November 23-25, ACM., 2015.

[B.03] MacKenzie S. B. The dangers of poor construct conceptualization. Journal of the Academy of Marketing Science 31, 3 (2003), 323–326., 2003.

[Bel07] A. B. Belgraver. The use of workshops for requirements engineering, 2007.

[BLCD23] R. Bougacha, R. Laleau, and S. Collart-Dutilleul. Formal alignment of requirements models with high-level architecture models. 27th International Conference on Engineering of Complex Computer Systems (ICECCS), Toulouse, France, 2023, pp. 216-225, doi: 10.1109/ICECCS59891.2023.00034, 2023.

[BS07a] Kitchenham B. and Charters S. Guidelines for performing systematic literature reviews in software engineering. Version 2.3. Tech. Rep., Keele University, 2007.

[BS07b] Kitchenham B. and Charters S. Guidelines for performing systematic literature reviews in software engineering. Version 2.3. Tech. Rep., Keele University, 2007.

[BS19] Shreya Banerjee and Anirban Sarkar. Domain-specific requirements analysis framework: ontology-driven approach. International Journal of Computers and Applications, 44:1, 23-47, 2019.

[CSL+01] Par Carlshamre, Kristian Sandahl, Mikael Lindvall, Bjorn Regnell, and Johan Natt och Dag. An industrial survey of requirements interdependencies in software product release planning. Proceedings. Fifth IEEE International Symposium on (pp. 84–91). IEEE., 2001.

[DFH16] Fabiano Dalpiaz, Xavier Franch, and Jennifer Horkoff. istar 2.0 language guide, 2016.

[EG04] Alexander Egyed and Paul Grünbacher. Identifying requirements conflicts and cooperation: How quality attributes and automated traceability can help. IEEE Software, vol. 21, no. 6, pp. 50-58, 2004.

[FNT14] F. Fellir, K. Nafil, and R. Touahni. System requirements prioritization based on ahp. Third IEEE International Colloquium in Information Science and Technology (CIST), Tetouan, Morocco, 2014, pp. 163-167, doi: 10.1109/CIST.2014.7016612., 2014.

[GG19] Ankita Gupta and Chetna Gupta. A novel collaborative requirement prioritization approach to handle priority vagueness and inter-relationships. Jaypee Institute of Information Technology, A-10 Sector 62, Noida, India, 2019.

[GP05] Dahlstedt Asa G. and Anne Persson. Requirements interdependencies: state of the art and future challenges. Engineering and managing software requirements (2005): 95-116, 2005.

[HGC21] Guan H., Cai G., and Zhao C. An automatic approach to extracting requirement dependencies based on semantic web. 8th International Conference on Dependable Systems and Their Applications (DSA), 2021.

[HLLZ08] Philipp Heim, Steffen Lohmann, Kim Lauenroth, and Jürgen Ziegler. Graph-based visualization of requirements relationships. Requirements Engineering Visualization, 2008. REV '08., 2008.

[IB01] HOH IN and BARRY W. BOEHM. Using winwin quality requirements management tools: A case study. Annals of Software Engineering 11, 141–174, 2001.

[JH08] Michael C. Jaeger and Anne Hoffmann. Assessing relations between non-functional requirements. Software Engineering 2008 - Workshopband, Fachtagung des GI-Fachbereichs Softwaretechnik, 18.-22.2.2008 in München, 2008.

[KBV12] Vishwajeet Kulshreshtha, John Boardman, and Dinesh Verma. The emergence of requirements networks: the case for requirements interdependencies. Int. J. Computer Applications in Technology, Vol. 45, No. 1, 2012.

[Kit04] B. Kitchenham. Procedures for performing systematic reviews. Keele, UK, Keele University 33, TR/SE-0401, 2004.

[MAL+20] John Mylopoulos, Daniel Amyot, Luigi Logrippo, Alireza Parvizimosaed, and Sepehr Sharifi. Social dependence relationships in requirements engineering. International i* Workshop, 2020.

[MCG19] Gonzalo Garcia Martinez, Alvaro Fernandez Del Carpio, and Luis Nunez Gomez. A model for detecting conflicts and dependencies in non-functional requirements. XLV Latin American Computing Conference (CLEI) 978-1-7281-5574, 2019.

[MMF+ay] Gharib M., Falco M., Nijboer F., Tinga A.M., D'Agostini S., Rovini E., Fiorini L., Cavallo F., and Taveter K. Dealing with emotional requirements for software ecosystems: Findings and lessons learned in the phara-on project. International Conference on Research Challenges in Information Science (pp. 99-114). Cham: Springer Nature Switzerland., 2024 May.

[Moh16] Gharib Mohamad. Privacy requirements: findings and lessons learned in developing a privacy platform. International Requirements Engineering Conference (RE). IEEE, 2016., 2016.

[MPLL+15] Tim Miller, Sonja Pedell, Antonio A. Lopez-Lorca, Antonette Mendoza, Leon Sterling, and Alen Keirnan. Emotion-led modelling for people-oriented requirements engineering: the case study of emergency systems. Journal of Systems and Software 105, 2015.

[MSMS14] Tabassum M., Siddik S., Shoyaib M., and Khaled S.M. Determining interdependency among non-functional requirements to reduce conflict. 3rd INTERNATIONAL CONFERENCE ON INFORMATICS, ELECTRONICS VISION, 2014.

[MVM00] L. Monostori, Zs. J. Viharos, and S. Markos. Satisfying various requirements in different levels and stages of machining using one general ann-based process model. Journal of Materials Processing Technology Volume 107, Issues 1–3, 22 November 2000, Pages 228-235, 2000.

[NFRN23] Noviyanto, Fiftin, Rozilawati Razali, and Mohd Zakree Ahmad Nazree. Understanding requirements dependency in requirements prioritization: a systematic literature review. International Journal of Advances in Intelligent Informatics 9.2 (2023): 249-272, 2023.

[NPV03] Robinson William N., Suzanne D. Pawlowski, and Vecheslav Volkov. Requirements interaction management. ACM Computing Surveys (CSUR) 35.2 (2003): 132-190, 2003.

[PM09] Runeson P. and Höst M. Guidelines for conducting and reporting case study research in software engineering. Empirical Software Engineering 14, 2 (2009), 131–164., 2009.

[RBA10] Svensson R., Regnell B., and Aurum A. Towards modeling guidelines for capturing the cost of improving software product quality in release planning. Proceedings of the 11th International Conference on Product Focused Software, pp.20-23, 2010.

[SAC19] Sumesh S., Krishna A., and Subramanian C. Game theory-based reasoning of opposing non-functional requirements using inter-actor dependencies. The Computer Journal 62(11):1557-1583, 2019.

[SCW+09] A. Sardinha, R. Chitchyan, N. Weston, P. Greenwood, and A. Rashid. Ea-analyzer: Automating conflict detection in aspect-oriented requirements. IEEE/ACM International Conference on Automated Software Engineering, Auckland, New Zealand, 2009, pp. 530-534, doi: 10.1109/ASE.2009.31, 2009.

[SHSM14] Safeeullah Soomro, Abdul Hafeez, Asadullah Shaikh, and Syed Hyder Abbas Musavi. Ontology-based requirement interdependency representation and visualization. International Multi Topic Conference, 2014.

[SJ05] Vasilache S. and Tanaka J. Bridging the gap between analysis and design using dependency diagrams. Third ACIS Int'l Conference on Software Engineering Research, Management and Applications (SERA'05), 2005.

[SK20] Sreenithya Sumesh and Aneesh Krishna. Sensitivity analysis of conflicting goals in the i* goal model. The Computer Journal 65(6), 2020.

[TSDL18] Zhixiang Tong, Xiaohong Su, Xiao Ding, and Jiaxin Lin. Optimizing software development requirements based on dependency relations. Journal of Information Hiding and Multimedia Signal Processing Ubiquitous International ISSN 2073-4212, 2018.

[vL01] Axel van Lamsweerde. Goal-oriented requirements engineering: A guided tour. Proceedings Fifth IEEE International Symposium on Requirements Engineering, 2001.

[Wie03] K.E. Wiegers. Software requirements, second edition. Microsoft Press, 2003.

[WLX21] Guo W., Zhang L., and Lian X. Putting software requirements under the microscope: automated extraction of their semantic elements. IEEE 29th International Requirements Engineering Conference (RE), 2021.

[YPSP20] Cheah Huei Yoong, Venkata Reddy Palleti, Arlindo Silva, and Christopher M. Poskitt. Towards systematically deriving defence mechanisms from functional requirements of cyber-physical systems. ASIA CCS '20: The 15th ACM Asia Conference on Computer and Communications Security, 2020.

Appendix

I. Table of selected papers

The following table Table 7 provides a comprehensive list of all the academic papers and sources that were reviewed during this research. A unique number identifies each entry in the table and includes the corresponding citation key, the title of the paper, the names of the authors, and the publication date. This table serves as a reference for all the literature that informed the development of my taxonomy, offering a detailed overview of the foundational works that contributed to the research.

N	Citation key	Paper name	Authors	Date
1	[CSL+01]	An Industrial Survey of Requirements Interdependencies in Software Product Release Planning	Par Carlshamre, Kristian Sandahl, Mikael Lindvall, Bjorn Regnell and Johan Natt och Dag	2001
2	[KBV12]	The emergence of requirements networks: the case for requirements inter-dependencies	Vishwajeet Kulshreshtha, John Boardman and Dinesh Verma	2012
3	[AAN15]	A Probability Algorithm for Requirement Selection In Component-Based Software Development	Ruba Alzyoudi, Khaled Almakadmeh and Hutaf Natoureah	2015
4	[RBA10]	Towards Modeling Guidelines for Capturing the Cost of Improving Software Product Quality in Release Planning	Svensson R., Regnell B. and Aurum A.	2010
5	[SAC19]	Game Theory-Based Reasoning of Opposing Non-functional Requirements using Inter-actor Dependencies	Sumesh S., Krishna A. and Subramanian C.	2019
6	[MSMS14]	Determining Interdependency Among Non-functional Requirements to Reduce Conflict	Tabassum M., Siddik S. and Shoyaib M. and Khaled S.M.	2014
7	[HGC21]	An Automatic Approach to Extracting Requirement Dependencies based on Semantic Web	Guan H., Cai G. and Zhao C.	2021

N	Citation key	Paper name	Authors	Date
8	[WLX21]	Putting software requirements under the microscope: automated extraction of their semantic elements	Guo W., Zhang L. and Lian X.	2021
9	[SJ05]	Bridging the Gap between Analysis and Design Using Dependency Diagrams	Vasilache S. and Tanaka J.	2005
10	[IB01]	Using WinWin Quality Requirements Management Tools: A Case Study	HOH IN and BARRY W. BOEHM	2001
11	[GG19]	A novel collaborative requirement prioritization approach to handle priority vagueness and inter-relationships	Ankita Gupta and Chetna Gupta	2019
12	[MCG19]	A Model for Detecting Conflicts and Dependencies in Non-Functional Requirements Using Scenarios and Use Cases	Gonzalo Garcia Martinez, Alvaro Fernandez Del Carpio and Luis Nunez Gomez	2019
13	[SHSM14]	Ontology-Based Requirement Interdependency Representation and Visualization	Safeeullah Soomro, Abdul Hafeez, Asadullah Shaikh, Syed Hyder Abbas Musavi	2014
14	[JH08]	Assessing Relations between Non-Functional Requirements	Michael C. Jaeger and Anne Hoffmann	2008
15	[EG04]	Identifying Requirements Conflicts and Cooperation: How Quality Attributes and Automated Traceability Can Help	Alexander Egyed and Paul Grünbacher	2004
16	[SK20]	Sensitivity Analysis of Conflicting Goals in the i* Goal Model	Sreenithya Sumesh and Aneesh Krishna	2020
17	[YPSP20]	Towards Systematically Deriving Defence Mechanisms from Functional Requirements of Cyber-Physical Systems	Cheah Huei Yoong, Venkata Reddy Palleti, Arlindo Silva and Christopher M. Poskitt	2020

N	Citation key	Paper name	Authors	Date
18	[MVM00]	Satisfying various requirements in different levels and stages of machining using one general ANN-based process model	L. Monostori, Zs. J. Viharos and S. Markos	2000
19	[BS19]	Domain-specific requirements analysis framework: ontology-driven approach	Shreya Banerjee and Anirban Sarkar	2019
20	[TSDL18]	Optimizing software development requirements based on dependency relations	Zhixiang Tong and Xiaohong Su and Xiao Ding and Jiaxin Lin	2018
21	[MAL+20]	Social Dependence Relationships in Requirements Engineering	John Mylopoulos, Daniel Amyot, Luigi Logrippo, Alireza Parvizimosaed and Sepehr Sharifi	2020
22	[DFH16]	iStar 2.0 Language Guide	Fabiano Dalpiaz, Xavier Franch and Jennifer Horkoff	2016
23	[vL01]	Goal-Oriented Requirements Engineering: A Guided Tour	Axel van Lamsweerde	2001
24	[MPLL+15]	Emotion-led modelling for people-oriented requirements engineering: the case study of emergency systems	Tim Miller, Sonja Pedell, Antonio A. Lopez-Lorca, Antonette Mendoza, Leon Sterling and Alen Keirnan	2015
25	[GP05]	Requirements interdependencies: state of the art and future challenges.	Dahlstedt Asa G. and Anne Persson	2005
26	[HLLZ08]	Graph-based Visualization of Requirements Relationships	Philipp Heim, Steffen Lohmann, Kim Lauenroth and Jürgen Ziegler	2008
27	[NFRN23]	Understanding requirements dependency in requirements prioritization: a systematic literature review.	Noviyanto and Fiftin and Rozilawati Razali and Mohd Zakree Ahmad Nazree	2023

N	Citation key	Paper name	Authors	Date
28	[SCW^{+}09]	EA-Analyzer: Automating Conflict Detection in Aspect-Oriented Requirements	A. Sardinha, R. Chitchyan, N. Weston, P. Greenwood and A. Rashid	2009
29	[BLCD23]	Formal alignment of requirements models with high-level architecture models	R. Bougacha, R. Laleau and S. Collart-Dutilleul	2023
30	[FNT14]	System requirements prioritization based on AHP	F. Fellir, K. Nafil and R. Touahni	2014

Table 7. Table of Researched Papers

II. Licence

Non-exclusive licence to reproduce thesis and make thesis public

I, **Elvin Mirzazada**,

1. herewith grant the University of Tartu a free permit (non-exclusive licence) to

   reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright,

   Towards a Novel Taxonomy for Requirements Interdependencies,

   supervised by Mohamad Gharib, Ph.D.

2. I grant the University of Tartu a permit to make the work specified in p. 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons licence CC BY NC ND 3.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.

3. I am aware of the fact that the author retains the rights specified in p. 1 and 2.

4. I certify that granting the non-exclusive licence does not infringe other persons' intellectual property rights or rights arising from the personal data protection legislation.

Elvin Mirzazada
12/08/2024