Baltic J. Modern Computing, Vol. 13 (2025), No. 4, pp. 834–861
http://doi.org/10.22364/bjmc.2025.13.4.05

ReInTa: A Novel Requirements Interdependencies Taxonomy

Mohamad GHARIB[1], Elvin MIRZAZADA[1]

University of Tartu, Estonia

{mohamad.gharib,elvin.mirzazada}@ut.ee

ORCID 0000-0003-2286-2819, ORCID 0009-0003-0195-579X

Abstract. Requirements interdependencies define how requirements relate to and influence one another, playing a critical role in the delivery of robust software systems. Despite their importance, existing literature lacks a unified framework to systematically capture and classify these interdependencies. This paper bridges this gap by introducing ReInTa, a novel taxonomy of requirements interdependencies derived through a Systematic Literature Review (SLR). ReInTa organizes the identified 16 interdependencies into three categories — Intra-type, Cross-type, and Multi-type—addressing gaps in fragmentation and terminology inconsistency. The taxonomy is evaluated against established frameworks, demonstrating broader coverage, including previously under-addressed interdependencies like Continuance, Cooperation, and iCost. A demonstration using an example abstracted from the PHArA-ON project illustrates RelnTa's practical utility in enhancing the Motivational Goal Model (MGM), enabling richer modeling of interactions among functional, quality, and emotional goals. ReInTa provides researchers and practitioners with a standardized and actionable framework for managing complex requirement relationships, ultimately improving system quality and development efficiency.

Keywords: Requirements interdependencies, Requirements taxonomy, Requirements relationships, Requirements modeling, Requirements Engineering

1 Introduction

The Requirements Engineering (RE) community widely acknowledges that most requirements are interrelated Dahlstedt and Persson (2005). These interdependencies define how requirements influence and relate to one another Tabassum et al. (2014). Empirical studies, such as those by Carlshamre et al. Carlshamre et al. (2001), suggest that only a fifth of the requirements are not related to or influenced by any other requirements. Given their importance, requirements interdependencies must be carefully managed Tabassum et al. (2014), as they significantly impact software quality, development

time, and cost Robinson et al. (2003). Failure to properly address these relationships can even contribute to software project failures Noviyanto et al. (2023).

Prior research has proposed diverse approaches for capturing requirements interdependencies (e.g., Dalpiaz et al. (2016); Van Lamsweerde (2001); Miller et al. (2015)) and identified various dependency types, such as requires, refines, similar, and conflicts Gharib et al. (2016). However, these contributions remain fragmented and lack standardization, revealing a critical gap: no unified framework exists to integrate key requirements interdependencies. This fragmentation is further complicated by the absence of standardized terminology and classification criteria Robinson et al. (2003). This complicates the comparison and synthesis of interdependencies, a fundamental aspect of software development. Consequently, researchers lack a common understanding and terminology, leading to potential analytical oversights and incorrect requirements analysis Dahlstedt and Persson (2005). Without a comprehensive, structured taxonomy, the identification, understanding, and application of requirements interdependencies continues to present substantial challenges.

The main objective of this paper is to develop a novel taxonomy for requirements interdependencies that offers a deeper and more comprehensive understanding of these relationships. The taxonomy will enable improved comprehension, analysis, and management of complex interdependencies in modern software systems. To achieve this objective, we have defined the following Research Questions (RQs):

RQ1: What types of requirements interdependencies/relationships exist in the literature?

RQ2: How are interdependencies between requirements utilized across different requirement types?

RQ3: What are the key coverage gaps in existing studies on requirements interdependencies?

This study employs a Systematic Literature Review (SLR) methodology to comprehensively survey existing research and identify the most mature and relevant studies in the field. Through careful analysis of these selected works, we address our defined RQs. Building upon these findings, we develop **ReInTa** (**Re**quirements **In**terdependency **Ta**xonomy), a novel taxonomy designed to systematically identify, classify, and manage requirements interdependencies. As we demonstrate in later sections, ReInTa has significant potential to enhance both software quality and development efficiency by providing structured guidance for handling complex requirement interdependencies.

The rest of this paper is structured as follows: Section 2 establishes the research baseline, followed by the methodology in Section 3. Section 4 presents our Systematic Literature Review (SLR), with results and findings discussed in Section 5. We introduce ReInTa and detail its construction in Section 6. The taxonomy evaluation appears in Section 7, followed by its application to a Motivational Goal Model (MGM) example in Section 8. Section 9 examines threats to validity, while Section 10 concludes the paper and outlines future work.

2 Baseline

Software requirements have evolved through distinct historical stages. Initially, the focus was predominantly on defining software system requirements—the precise specifications of what the software must do Greenspan et al. (1982). By the 1980s, this perspective broadened to explicitly include the operational environment, acknowledging that a system's functionality depends upon its interactions with external entities and surrounding conditions Dubois et al. (1986). An important shift occurred in the 1990s with the incorporation of the social and organizational context, recognizing that software is embedded within complex human structures, goals, and norms Yu (1993). This modern, holistic view ensures that the developed software system is not only technically sound but also viable, valuable, and effectively integrated within its intended ecosystem of stakeholders, policies, and cultural practices.

Requirements do not exist in isolation; they often depend on and influence one another Dahlstedt and Persson (2005). These interdependencies can take several forms, including: *Functional dependencies*, where fulfilling one requirement depends on functionality provided by another Zave and Jackson (1997); Dalpiaz et al. (2016); Van Lamsweerde (2001); Miller et al. (2015); *Temporal dependencies*, where one requirement must be fulfilled before another can be addressed Van Lamsweerde et al. (1995); *Resource dependencies*, where fulfilling one requirement depends on resources provided by another Dalpiaz et al. (2016); and *Conflict dependencies*, where fulfilling one requirement may hinder another Nuseibeh et al. (2000); Dalpiaz et al. (2016).

Capturing and analyzing requirements interdependencies is critical for several reasons, most notably: *Impact analysis*, understanding dependencies helps assess how changes to one requirement may propagate through the system Arnold and Bohner (1996). *Risk mitigation*, unrecognized dependencies can lead to design flaws, integration failures, or project delays Wiegers and Beatty (2013). *Prioritization and trade-offs*, dependency mapping supports informed decision-making when balancing competing constraints Helfert and Herrmann (2002). Finally, *traceability,* clear interdependencies enhance requirements traceability and ensure alignment with system architecture and test cases Dahlstedt and Persson (2005).

Requirements interdependencies play a critical role throughout the entire requirements engineering lifecycle, influencing elicitation, classification, prioritization, and validation phases. Recognizing and managing these interdependencies is therefore essential for effective requirements engineering. In what follows, we discuss each of these phases, followed by how requirements interdependencies are relevant to them.

Requirements elicitation aims at discovering the requirements for the system-to-be through consulting relevant stakeholders, investigating system documentation, and applying domain knowledge Sommerville and Sawyer (1997); Kotonya et al. (1998); Sommerville (2007). During this process, dependencies naturally emerge as stakeholders articulate needs that may conflict or rely on one another Zave and Jackson (1997), necessitating specialized techniques like scenario analysis to uncover these implicit relationships Dahlstedt and Persson (2005). The identification of such dependencies at this early stage proves critical for subsequent development phases.

Requirements classification transforms an unstructured collection of requirements into coherent clusters Sommerville (2007). While multiple classification approaches ex-

ist Aurum and Wohlin (2005), the fundamental distinction between functional requirements (specifying what the system shall do) and non-functional requirements (defining how it should operate) remains paramount Chung et al. (1999). As classification proceeds, interdependencies play a key role in determining appropriate grouping strategies, ensuring logical structuring while preventing artificial fragmentation of related requirements Bass et al. (1997). This facilitates more effective requirements management throughout the development lifecycle.

Requirements prioritization involves ranking requirements based on their relative importance Sommerville (2007); Helfert and Herrmann (2002), guiding decisions about implementation sequencing. This process must carefully consider requirements interdependencies Karlsson (2004), as high-priority features may depend on lower-priority foundational elements Berander (2004). Methods like dependency-aware ranking have been developed to optimize resource allocation while respecting these critical relationships Leffingwell and Widrig (2000).

Finally, *requirements validation* ensures that the final requirements set accurately reflects stakeholder expectations Kotonya et al. (1998); Sommerville (2007). At this stage, interdependencies become particularly important as they determine whether testing should evaluate requirements in isolation or as integrated components. Proper attention to these relationships helps prevent system failures caused by overlooked interactions between requirements Nuseibeh et al. (2000). Please note that requirements validation ensures a requirement set is correct, complete, and consistent with stakeholder expectations, typically through five checks: validity, completeness, consistency, realism, and verifiability Sommerville (2007); Terry Bahill and Henderson (2005). Among these, interdependencies are central to consistency checks (e.g., identifying conflict or exclusion relationships) and completeness checks (e.g., revealing missing prerequisites). They also support realism assessments by exposing cost or technical constraints and aid verifiability by establishing traceable links (e.g., satisfies) for testing. Thus, analyzing interdependencies is a core mechanism, not an ancillary activity, for robust validation.

In summary, requirements interdependencies form a critical dimension throughout the entire RE lifecycle. Their proper identification and management directly contribute to more robust systems, informed decision-making, and ultimately project success. As the discussed phases demonstrate, neglecting these relationships risks compromising system integrity, while their systematic consideration enables more effective requirements engineering practices.

3 Research Methodology

This study follows a three-phase, iterative research methodology, as illustrated in Figure 1. The process begins with a Systematic Literature Review (SLR) structured around three sub-phases, each corresponding to one of our Research Questions (RQs). The second phase synthesizes these findings to develop our taxonomy. Finally, we evaluate and refine the taxonomy. In what follows, the three key phases of the methodology are briefly discussed.

1. **Identify the most relevant studies to answer the RQs via an SLR:** This phase systematically identifies the most mature and relevant studies on requirements inter-

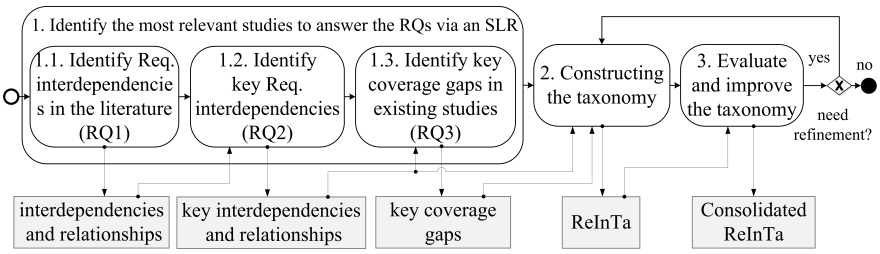

Fig. 1. The methodology process

dependencies to address the RQs. It follows established SLR protocols, including: (1) planning the review, (2) defining search strategies and study selection criteria, and (3) executing data extraction and synthesis. The SLR outcomes provide input for addressing RQ1, RQ2, and RQ3 through three sub-phases that are dedicated to them, as detailed in Sections 4 and 5.

2. **Constructing the taxonomy:** This phase systematically develops the taxonomy by synthesizing two key outputs from the SLR: (1) the identified requirements interdependencies (from RQ2) and (2) the coverage gaps in existing studies (from RQ3). The construction process involves three critical activities: first, identifying and rigorously defining core concepts for capturing requirements interdependencies; second, resolving synonymous terms to ensure conceptual clarity and distinctness; and third, categorizing these refined concepts into logical, high-level groupings that form the structural foundation of the taxonomy. The taxonomy construction is discussed in Section 6.
3. **Evaluate and improve the taxonomy:** This phase evaluates the resulting taxonomy, assessing its internal coherence and consistency, its completeness regarding existing work, and its practical utility. The evaluation involved a comparative analysis of the developed taxonomy against three established requirements engineering frameworks: iStar 2.0 Dalpiaz et al. (2016), KAOS (Knowledge Acquisition in automated specification) Van Lamsweerde (2001), and Emotion-led Modeling language (Motivational Goal Model (MGM)) Miller et al. (2015). This comparison allowed for an assessment of the coverage of the taxonomy and its alignment with existing conceptualizations, leading to minor refinements. To demonstrate the practical applicability of the taxonomy, we applied it to model relationships within a real-world example abstracted from the PHArA-ON (Pilots for Healthy and Active Ageing in Europe) H2020 European Project[1]. This proves the practical relevance of the taxonomy and its ability to represent a wider range of interdependencies compared to existing approaches. The evaluation and demonstration of the taxonomy are presented in Sections 7 and 8, respectively.

Through this systematic, iterative methodology, we ensure the resulting taxonomy achieves three critical objectives: (1) strong theoretical foundations grounded in comprehensive literature analysis, (2) direct mitigation of identified knowledge gaps in cur-

[1] `https://www.pharaon.eu/`

rent requirements engineering practice, and (3) demonstrable practical utility. This guarantees that the final taxonomy is both theoretically sound and practically relevant for improving requirements engineering activities.

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

To systematically identify the most relevant and mature studies on requirements interdependencies, we conducted a Systematic Literature Review (SLR) following the established guidelines by Kitchenham (2004). As illustrated in Figure 2, our SLR methodology comprises three distinct phases:

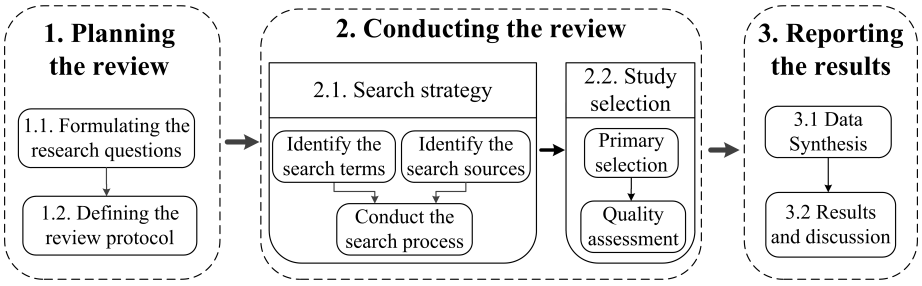

Fig. 2. The Systematic Literature Review Process.

1. **Planning the review:** We formulated the RQs and developed a comprehensive review protocol to ensure methodological rigor.
2. **Conducting the review:** This phase involved (a) identifying appropriate search terms and literature sources, (b) executing the search process, and (c) systematically selecting relevant studies through predefined criteria.
3. **Reporting the results of the review:** We extracted and analyzed detailed information from the selected studies to address each RQ.

The subsequent sections elaborate on each phase in detail.

1. Planning the review. This phase is fundamental for the success of the review, as it establishes the research objectives and the process to conduct the review. This phase encompasses three primary activities:

1.1. Formulating the RQs is a vital activity since RQs serve as the basis for developing the overall approach for a systematic review Kitchenham (2004). Therefore, we formulate three RQs of this research (presented earlier) to identify key requirements interdependencies that have been presented in the literature, as well as the gaps in existing studies on such interdependencies.

1.2. 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, we discuss how each of these activities was performed.

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.

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

Identify the search terms. We derived the main search terms from the research questions. In particular, we used the Boolean AND to link the major terms, and we 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. We 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. We used the search terms to search the selected electronic database sources, and all returned studies were considered.

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 papers for WoS, and 59 papers for ACM Digital Library, resulting in 1787 papers. After removing duplicate papers, we had 1485 papers. We read the titles and abstracts and skimmed through the rest of the papers to apply the inclusion and exclusion criteria (shown in Table 1). This comprehensive approach ensured that the final selection of papers was highly relevant and contributed to answering the RQs. This resulted in selecting 106 papers.

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.

The 106 papers were fully read, and only papers that contained sufficient information to contribute to satisfying at least three of the quality assessment questions (shown in Table 2) were included. More specifically, the four quality assessment questions were

Table 2. Quality Assessment Questions.

QA Question
Q1. Are the objectives of the proposed work related to eliciting, capturing, modeling, and/or analyzing requirements interdependencies?
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 justified by appropriate examples?

used to filter 106 papers, requiring studies to be directly relevant to requirements interdependencies (Q1), offer clear definitions (Q2), provide sufficient conceptual depth (Q3), and, preferably, include practical validation or examples (Q4). This resulted in the selection of 22 papers. A simplified representation of the search and selection process is shown in Figure 3. For transparency and further research, a spreadsheet documenting the lists of the selected papers in each step has been made available online[2].

3. Reporting the results. This phase involves summarizing the results, and it consists of two main activities: 1- data synthesis; and 2- results and discussion.

Data synthesis aims at combining the findings of the selected studies in a way that allows answering the RQs. To facilitate this activity, each of the selected papers has been analyzed, and its contribution to the RQs has been summarized. In what follows, we describe how data synthesis was conducted. Data related to *RQ1* were extracted from the list of selected papers, which were analyzed to identify relevant relationships/interdependencies that are shown in Table 3 along with their frequency of appearance in the selected papers. Key relationships/interdependencies are distinguished using **Bold** typeset. These key relationships/interdependencies have been selected based on their frequency of appearance and their importance for capturing relationships/interdependencies among requirements. To answer *RQ2*, we analyzed how each identified key requirement interdependency has been used in the relevant literature. Table 4 includes these key relationships/interdependencies accompanied by a brief description of each of them. Finally, to answer *RQ3*, we first categorized interdependencies into three groups. Then, categorized the selected studies based on their coverage of these three groups (see Table 5). In the following section, we discuss how each *RQ* has been answered.

5 Review results and discussion

In this section, we present and discuss the findings of this review concerning each RQ in its corresponding step, as follows:

RQ1: What types of requirements interdependencies/relationships exist in the literature? To address this RQ, we systematically analyzed the selected literature to identify the most relevant and frequently cited requirements interdependencies. Our analysis

[2] `https://doi.org/10.5281/zenodo.16731599`.

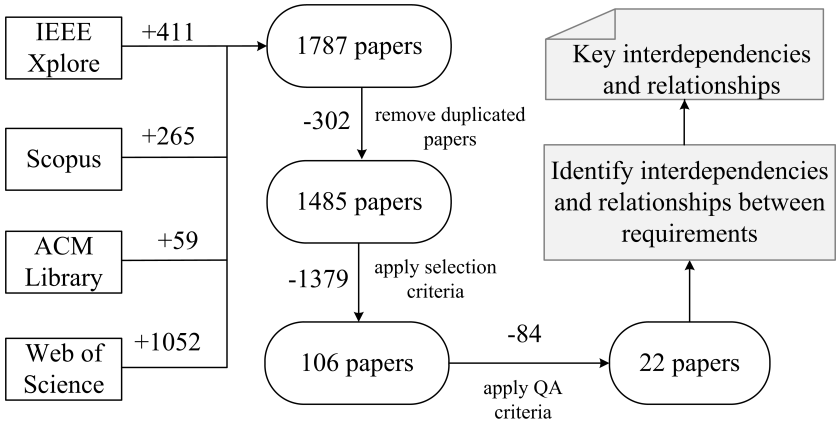

Fig. 3. Paper search and selection process.

revealed 30 interdependencies (shown in Table 3), from which we selected 16 key relationships highlighted in **bold**. The selection criteria were based on: (1) conceptual distinctiveness (removing overlaps with similar interdependencies), (2) frequency of appearance across studies, and (3) their critical role in capturing essential requirement relationships. This selection process ensures our findings focus on the most impactful interdependencies for requirements engineering practice.

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 (FR), while others capture relationships between Non-Functional Requirements (NFR). The first type of relationship captures the interdependency between what a system must do, and the last captures the relationships among qualities the system strives to achieve. Moreover, some identified dependencies involve an interaction between FR and NFR. This type of relation highlights the interdependency between what a system must do and the qualities or constraints under which the system must operate. Moreover, there were some infrequently identified dependencies such as *Resource-Based Dependencies*, which considers the availability of resources that a certain requirement is dependent on the completion or availability of resources allocated to other requirements.

RQ2: How are interdependencies between requirements utilized across different requirement types? Interdependencies between different requirement types—specifically FRs and NFRs—play an important role in ensuring that the overall system design and development are coherent, efficient, and aligned with both users' needs and system constraints. Through examining the selected papers, we refined and precisely characterized these interdependencies, establishing an unambiguous context of their use that articulates their distinct functions within system specification. Our analysis of how these interdependencies have been employed and defined across the papers enabled us to: (1) disambiguate their conceptual boundaries, (2) clarify their operational roles, and (3) standardize their interpretation. The complete list of key interdependencies is presented

Table 3. Requirements dependencies in the selected papers

	Carlshamre et al. (2001)	**Kulshreshtha et al. (2012)**	**Alzyoudi et al. (2015)**	**Svensson et al. (2010)**	**Sumesh et al. (2019)**	**Tabassum et al. (2014)**	**Guan et al. (2021)**	**Guo et al. (2021)**	**Vasilache and Tanaka (2005)**	**In and Boehm (2001)**	**Gupta and Gupta (2022)**	**Martinez et al. (2019)**	**Soomro et al. (2013)**	**Jaeger and Hoffmann (2008)**	**Egyed and Grünbacher (2004)**	**Sumesh and Krishna (2022)**	**Mylopoulos et al. (2020)**	**Dalpiaz et al. (2016)**	**Van Lamsweerde (2001)**	**Miller et al. (2015)**	**Tong et al. (2018)**	**Monostori et al. (2000)**
And	✓		✓	✓		✓	✓	✓			✓	✓			✓	✓		✓	✓	✓		
Temporal	✓																					
Replace								✓														
Or	✓		✓	✓		✓	✓	✓			✓	✓			✓	✓		✓	✓	✓		
Before			✓																	✓		
Compliance		✓														✓			✓			
Wish																			✓			
Contribution					✓											✓		✓	✓			
Continuance		✓																				
Non-Invertible																						✓
Consequential		✓																				
Conflict			✓						✓				✓		✓	✓			✓		✓	
Concurency									✓													
Cooperation		✓													✓				✓			
CValue	✓																					
Dependence													✓								✓	
ICost	✓					✓				✓												
Social																	✓					
Exclusion								✓	✓													
Requires	✓					✓		✓										✓		✓		
Coordinate													✓								✓	
Contradiction													✓								✓	
Contractual		✓																				
S-Cost										✓												
Cause-Effect									✓													
Time									✓													
Supports			✓					✓				✓							✓		✓	
Abstraction									✓													
Equivalence								✓														
Satisfies								✓										✓	✓			

in Table 4. Further, we have classified these interdependencies based on the type of the involved requirements into the following three top-level categories:

1. **Intra-type (FR-to-FR or NFR-to-NFR) interdependencies:** capture relationships among the system-to-be functionalities/capabilities (FR-to-FR) or among the system-to-be qualities (NFR-to-NFR). These include *(1) FR-to-FR interdependencies:* are used to capture the hierarchies and sequence relationships among the FR of the system-to-be (e.g., *And*, *Or*). *(2) NFR-to-NFR interdependencies:* are used to capture the relationships among the quality attributes of the system-to-be (e.g., *Cooperation*). *(3) FR-to-FR or NFR-to-NFR interdependencies:* capture interactions either among functional capabilities (FR-to-FR) or among quality attributes (NFR-to-NFR) of the system-to-be (e.g., *Conflict*, *Consequential*).
2. **Cross-type (FR-to-NFR and NFR-to-FR) interdependencies:** capture how the capabilities of the system-to-be might influence or get influenced by its qualities (e.g., *Exclusion - (FR-to-NFR)*, *Compliance - (NFR-to-FR)*, *Contribution (FR-to-NFR or NFR-to-FR)*), i.e., they capture how the capabilities of the system-to-be fundamentally intertwines with its qualities.

(3) Multi-type (FR/NFR-to-FR/NFR) interdependencies: describe interdependencies that can be both Intra-type (FR-to-FR or NFR-to-NFR) and cross-type (FR-to-NFR and NFR-to-FR) interdependencies (e.g., *Supports*, *ICost*). Unlike the first two categories, which are confined to either homogeneous (FR/FR, NFR/NFR) or bidirectional (FR/NFR) dependencies, multi-type interdependencies involve dynamic combinations of FR and NFR, often emerging in scenarios where system behaviors and qualities are co-dependent in non-linear ways (e.g., an FR modifying an NFR, which in turn impacts another NFR).

Table 4: Key requirements interdependencies in relevant literature

And refines top-level functional requirement (FR) into several lower-level FRs, where the fulfillment of the top-level FR requires the fulfillment of all lower-level FRs Carlshamre et al. (2001); Alzyoudi et al. (2015); Svensson et al. (2010); Tabassum et al. (2014); Guan et al. (2021); Guo et al. (2021); Gharib et al. (2021).
Or refines top-level FR into several lower-level FRs, where the fulfillment of the top-level FR require the fulfillment of any of the lower-level FRs Carlshamre et al. (2001); Alzyoudi et al. (2015); Svensson et al. (2010).
Before ensures that one FR must be fulfilled before another can be satisfied Alzyoudi et al. (2015); Van Lamsweerde (2001).
Contractual captures the relationship where one requirement necessitates data from another Kulshreshtha et al. (2012). As FR requires data in most existing languages, we limit this interdependency to FR-to-FR.

Cooperation captures a state when two or more requirements collaborate to achieve a shared goal, where the effective execution of these requirements enhances or enables the execution of another requirement Kulshreshtha et al. (2012); Egyed and Grünbacher (2004); Van Lamsweerde (2001). While literature does not explicitly limit cooperation to NFRs, we argue they are uniquely suited to NFRs due to their inherent traits: (1) their satisfaction involves continuous trade-offs rather than binary fulfillment (e.g., balancing security vs. usability), and (2) they produce emergent system qualities through synergy (e.g., reliability and performance jointly enabling resilience). Moreover, And, a FR-to-FR interdependency, can capture how two or more FRs can collaborate to achieve a top level FR.
Conflict captures a state where two requirements (FR-to-FR or NFR-to-NFR) are mutually incompatible, creating contradictions that must be resolved to maintain system coherence and functionality Soomro et al. (2013); Egyed and Grünbacher (2004). In iStar Dalpiaz et al. (2016) "break" between FR-to-NFR has been introduced. However, break is not equivalent to conflict conceptually.
Continuance captures a state where one requirement triggers another, ensuring a logical and orderly development process Kulshreshtha et al. (2012). **Continuance** is applicable to both FRs and NFRs.
Consequential captures a dependency between two requirements (FR or NFR), where modifying or fulfilling one directly necessitates a corresponding adjustment or fulfillment of the other (FR or NFR) to maintain consistency Kulshreshtha et al. (2012).
Compliance captures the conformity of one requirement (usually a FR) to another NFR concerning laws, rules, standards, or policies Kulshreshtha et al. (2012); Sumesh and Krishna (2022).
Exclusion captures a state when the implementation of one NFR is explicitly prohibited or invalidated by a FR, often due to mutual exclusivity or conflicting nature Vasilache and Tanaka (2005).
Contribution captures how requirements collectively enable or hinder the fulfillment of other requirements Sumesh et al. (2019). It operates bidirectionally between FRs and NFRs—capturing how FR implementations influence NFR satisfaction (e.g., caching improving performance) and how NFRs constrain FR (e.g., security mandating encryption). While iStar Dalpiaz et al. (2016) considers NFR-to-NFR contributions, we deliberately exclude these interactions because they are more accurately represented through other interdependency types: Cooperation for synergistic NFR pairs (e.g., scalability and availability enhancing system resilience) and Conflict for competing qualities (e.g., security versus usability).
Satisfies captures a relationship where the implementation of a requirement ensures the fulfillment of the specified objectives or criteria of another requirement Guo et al. (2021); Dalpiaz et al. (2016); Van Lamsweerde (2001), applicable to both FRs and NFRs.
Supports captures a relationship where the successful implementation of one requirement assists the effective achievement of another Alzyoudi et al. (2015); Van Lamsweerde (2001), applicable to both FRs and NFRs.

Requires captures a relationship where the fulfillment of one requirement (FR or NFR) is contingent upon the satisfaction of another requirement (FR or NFR), indicating that one requirement relies on another to function effectively Tabassum et al. (2014); Guo et al. (2021); Dalpiaz et al. (2016); Miller et al. (2015).
ICost captures the impact of implementing one requirement (FR or NFR) on the cost of implementing another requirement (FR or NFR), either increasing or decreasing the expense Carlshamre et al. (2001); Tabassum et al. (2014); In and Boehm (2001).
CValue captures the impact of implementing one requirement (FR or NFR) on the perceived value of another requirement (FR or NFR), influencing development prioritization and the overall value proposition of the system Carlshamre et al. (2001).

RQ3: What are the key coverage gaps in existing studies on requirements interdependencies? We answer this question by comparing the coverage of the selected papers with the three categories of requirements interdependencies. The results presented in Table 5 reveal important insights into how existing studies address the three categories of requirements interdependencies: Intra-type, Cross-type, and Multi-type. Most studies provide partial coverage of Intra-type interdependencies (FR-to-FR or NFR-to-NFR), reflecting their alignment with traditional requirements engineering practices like goal decomposition and conflict detection. For instance, only Carlshamre et al. Carlshamre et al. (2001), Dalpiaz et al. Dalpiaz et al. (2016), and Van Lamsweerde Van Lamsweerde (2001) sufficiently cover these homogeneous relationships. While some studies (e.g., In and Boehm (2001)) offer poor coverage, likely due to their narrower focus on specific interdependency types such as temporal or resource constraints.

Cross-type interdependencies (FR-to-NFR or NFR-to-FR) receive moderate attention in the papers. Despite this, nearly half of the reviewed papers overlook Cross-type relationships entirely. This gap suggests that many approaches still prioritize homogeneous requirement analysis over heterogeneous interactions, limiting their ability to address real-world scenarios where functionality and quality constraints intersect. Multi-type interdependencies (FR/NFR-to-FR/NFR) also received moderate attention, where only a handful of studies (e.g., Tong et al. (2018); Kulshreshtha et al. (2012)) sufficiently address these interdependencies. The scarcity of research in this area highlights a critical gap in requirements engineering, as Multi-type dependencies are essential for modeling dynamic systems where FR and NFR co-evolve. The limited coverage in existing frameworks like iStar Dalpiaz et al. (2016) and KAOS Van Lamsweerde (2001) further validates the need for the comprehensive taxonomy proposed in this paper.

The uneven distribution of coverage across these categories underscores the fragmented nature of current research on requirements interdependencies. While Intra-type relationships are well-established, the lack of standardized approaches for Cross-type and Multi-type dependencies hinders holistic requirements management.

6 Constructing the Taxonomy for Requirements Interdependencies

The construction of the taxonomy was straightforward, as we first identified key interdependencies while addressing *RQ1* (see Table 3) and later categorized them into three

Table 5. A coverage of interdependency categories in selected primary studies.

Paper	Intra-type interd.	Cross-type interd.	Multi-type interd.
Carlshamre et al. (2001)	●	◐	◐
Kulshreshtha et al. (2012)	◐	○	●
Alzyoudi et al. (2015)	◐	◐	◐
Svensson et al. (2010)	◐	◐	○
Sumesh et al. (2019)	◐	◐	○
Tabassum et al. (2014)	◐	○	○
Guan et al. (2021)	◐	○	◐
Guo et al. (2021)	◐	○	○
Vasilache and Tanaka (2005)	◐	○	●
In and Boehm (2001)	○	○	◐
Gupta and Gupta (2022)	◐	◐	◐
Martinez et al. (2019)	●	◐	◐
Soomro et al. (2013)	◐	○	○
Jaeger and Hoffmann (2008)	◐	○	○
Egyed and Grünbacher (2004)	◐	○	◐
Sumesh and Krishna (2022)	◐	◐	○
Mylopoulos et al. (2020)	◐	◐	○
Dalpiaz et al. (2016)	●	◐	◐
Van Lamsweerde (2001)	◐	◐	◐
Miller et al. (2015)	◐	○	○
Tong et al. (2018)	◐	◐	●
Monostori et al. (2000)	◐	○	○
● Sufficiently covered; ◐ Partially covered; ○ Poorly covered			

distinct groups in *RQ2* (see Table 4). The resulting taxonomy is illustrated in Figure 4, which highlights the top-level categories of requirements interdependencies. This section details each interdependency type, clarifies conceptual distinctions between them, and illustrates their application with relevant examples.

1. Intra-type (FR-to-FR or NFR-to-NFR) interdependencies.

And *interdependency* decomposes complex FR into operational sub-requirements. For example, "**FR1.** *The system shall process online payments"* can be refined into "**FR1.1.** *The system shall validate the user's credit card information"* and "**FR1.2.** *The system shall confirm payment completion"*. Fulfilling **FR1** demands the fulfillment of **FR1.1** and **FR1.2**.

Or *interdependency* provides alternative refinements for a FR. For example, "**FR1.** *The system shall authenticate users"* can be refined into "**FR1.1.** *The system shall authenticate users via user name and password"* and "**FR1.2.** *The system shall authenticate users via biometric verification (fingerprint/facial recognition)*". Fulfilling **FR1** demands the fulfillment of **FR1.1** or **FR1.2**, offering design flexibility.

Before *interdependency* ensure that a dependent requirement (**FR2**) cannot be implemented until its prerequisite requirement (**FR1**) has been satisfied. For example, *Before* can enforce the implementation of "**FR1.** *The system shall store user profile data in a database*" before "**FR2.** *The system shall display user profiles on the dashboard*".

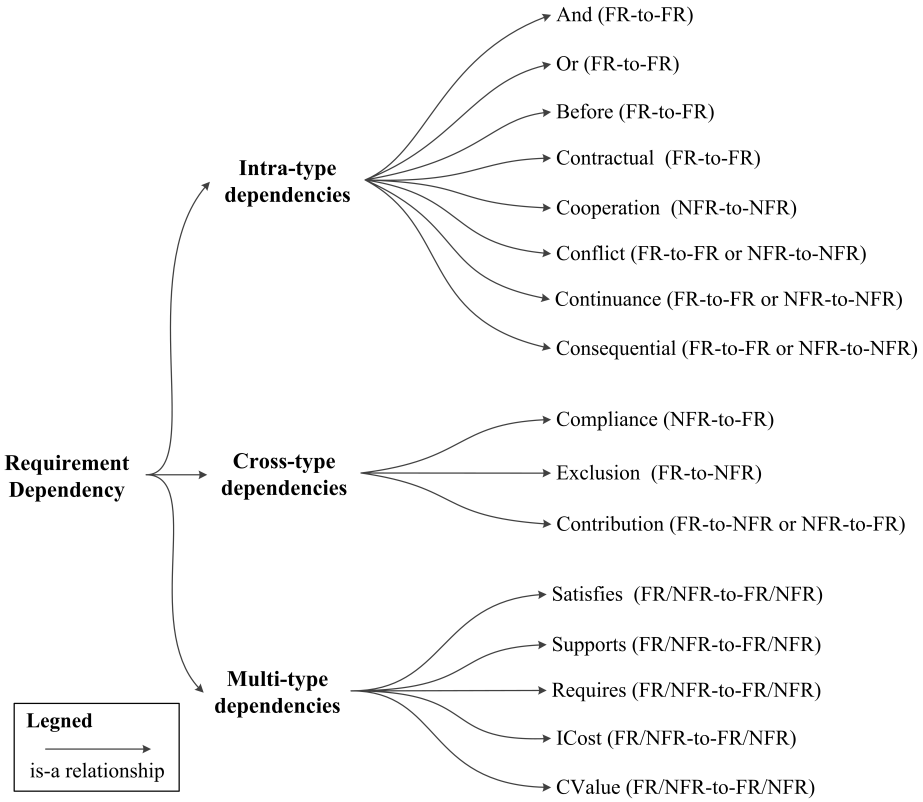

Fig. 4. The Proposed Novel Taxonomy of Requirements Interdependencies.

Contractual *interdependency* mandates data flow between FRs Kulshreshtha et al. (2012). Unlike *Before* that enforces implementation order, *Contractual* ensures data flow. For example we can use this interdependency to contractually obligate "**FR1.** *The payment system shall generate a transaction receipt containing: amount, date, and merchant ID"* to provide specific data to "**FR2.** *The accounting module shall record all transactions using the receipt data (amount, date, merchant ID)"*.

Cooperation *interdependency* becomes handy when two or more NFRs collaborate to achieve a shared goal. For instance, consider a cloud service where "**NFR1.** *Resilience: requires the system to withstand 10x traffic spikes"*, which can be achieved through the Cooperation of "**NFR2.** *Availability: mandating 99.9% uptime during 10x normal load"* and "**NFR3.** *Performance: requiring ¡2s API response under peak loads"*. **NFR2.** and **NFR3.** create a resilient system where the combined effect exceeds their individual contributions, exemplifying the emergent properties characteristic of NFR cooperation.

Conflict *interdependency* can arise in both FRs and NFRs. Among FRs, conflicts occur when implementation needs are mutually exclusive, creating binary incompatibilities that require prioritization or design arbitration. For example, "**FR1.** *The system*

shall delete all user-identifiable data 30 days after account termination (GDPR Article 17)" is clearly conflicting with "**FR2.** *The system shall retain complete transaction audit logs for one year"*. In contrast, NFR conflicts emerge from fundamental quality trade-offs. For example, "**NFR1.** *All communications shall use AES-256 encryption"* and "**NFR2.** *API endpoints shall respond within 100ms for 95% of requests"* are clearly conflicting and cannot be satisfied simultaneously.

Continuance *interdependency* can be applied to both FRs and NFRs, dictating the order of requirements and conditions for successful project completion. For example, "**FR1.** *The system shall allow users to add items to a shopping cart"* triggers "**FR2.** *The system shall calculate and display the total order cost in real-time"*. This ensures that developers cannot implement real-time price updates (**FR2.**) without first building the cart functionality (**FR1.**). Another example, "**NFR1.** *The system shall handle 10,000 concurrent users"* triggers "**NFR2.** *The database shall support connection pooling to manage high traffic.*". Unlike Before that specifies a temporal implementation constraint between FRs, Continuance describes a logical triggering relationship where fulfilling one requirement (e.g., "ensure data privacy") inherently necessitates addressing another (e.g., "implement encryption").

Consequential *interdependency* maintains consistency between requirements (FR or NFR) when one of them changes. For example, if "**FR1.** *The shopping cart shall apply location-based tax calculations during checkout."* is modified to "apply real-time tax API calls", then, "**FR2.** *The system shall store the user's verified shipping address before payment processing."* must consequentially be updated to: "Store shipping address with timestamp verification" (to ensure tax calculations use current rates). Unlike *Continuance*, which dictates a logical trigger for implementation order, *Consequential* ensures mutual adjustment during system evolution.

2. Cross-type (FR-to-NFR and NFR-to-FR) interdependencies.

Compliance *interdependency* guarantees that system implementation aligns with external standards (e.g., GDPR, ISO) or internal governance rules, mitigating legal and operational risks. For example, "**FR1.** *Doctors shall access patient medical histories."* must comply with "**NFR1.** *Patient data access shall comply with the GDPR"*.

Exclusion *interdependency* is essential in cases where specific functionalities are unable to coexist within some system qualities. For example, "**FR1.** *The car shall allow drivers to manually override autonomous controls at any time."* and "**NFR1.** *The system shall prevent all driver actions that could cause unsafe maneuvers."* **FR1.** requires unrestricted manual control, while **NFR1.** demands forced system intervention to block dangerous actions.

Contribution *interdependency* operates bidirectionally between FRs and NFRs, capturing how one requirement enables or hinders another. For example, implementing **FR1.** *The system shall compress image uploads using lossy compression (JPEG)."* positively contributes to "**NFR1.** *The system shall minimize bandwidth usage during file transfers."*.

(3) Multi-type (FR/NFR-to-FR/NFR) interdependencies.

Satisfies *interdependency* indicates that one requirement completely fulfills another objectives or criteria. For example, implementing **FR2.** *The steering control module shall implement triple redundant sensors with voting logic."* regarding an autonomous

vehicle control system satisfies "**NFR1.** *The system shall achieve ASIL-D fault tolerance per ISO 26262."* in an autonomous vehicle system.

Supports *interdependency* indicates partial assistance where one requirement improves another requirement achievement likelihood without guaranteeing it. Unlike *Satisfies* (which ensures complete fulfillment), *Supports* identifies probabilistic enhancements. Consider the following two requirements: "**NFR1.** *The system shall maintain* $\leq$*10ms latency for emergency braking signals."* and "**FR1.** *The controller area network (CAN) bus shall prioritize braking messages over non-critical signals."* **FR1.** supports **NFR1.** by improving response times, but cannot alone guarantee the $\leq$10ms target.

Requires *interdependency* captures essential logical dependencies where one requirement cannot function without another. Unlike *Satisfies* (fulfillment) or *Before* (sequence), *Requires* denotes prerequisite necessity. For example, "**FR1.** *The system shall activate emergency braking when obstacles are detected"* requires "**NFR1.** *The system shall process lidar sensor data in real-time"* to function.

ICost *interdependency* captures how one requirement affects another's implementation cost. For example, "**FR1.** *The system shall record HD video (1080p) from all cameras"* increases the storage cost of "**FR2.** *The system shall store video footage for 30 days.".*

CValue *interdependency* captures how one requirement enhances another's perceived value. For example, "**FR1.** *Let users save favorite restaurants"* increases the value of "**FR2.** *Show personalized recommendations."* by providing important user preference data.

In conclusion, the taxonomy presented here provides a structured set of interdependencies to model complex requirement relationships. It is important to note that these interdependencies are not isolated; they often interact and complement each other within a requirements model. For example, a high-level goal might be achieved not through a simple And/Or refinement, but through the combined effect of a functional requirement and a non-functional requirement, linked by a Satisfies or Requires Cross-type relationship. More complex chains can emerge: for instance, an And-refinement of a goal might lead to a sub-goal that is in Conflict with another requirement, while a separate branch of the refinement Contributes to a key quality goal. Similarly, realizing one requirement might increase the cost (ICost) of another, creating a trade-off. Therefore, the practical power of this taxonomy lies in its ability to be used compositionally, allowing practitioners to capture the intricate networks of dependencies where functional, qualitative, and emotional requirements continuously influence one another to define the complete system behavior.

7 Evaluation

To evaluate the comprehensiveness and relevance of our proposed taxonomy, we conducted a comparative analysis against three well-established requirements engineering frameworks: iStar 2.0 Dalpiaz et al. (2016), KAOS (Knowledge Acquisition in automated specification) Van Lamsweerde (2001), and Emotion-led Modeling language (Motivational Goal Model (MGM)) Miller et al. (2015). The frameworks selected for evaluation—iStar 2.0, KAOS, and MGM—were chosen to provide a comprehensive

assessment across the principal paradigms of goal-oriented requirements engineering. This selection ensures the taxonomy is evaluated against diverse and complementary conceptualizations of requirements relationships. iStar 2.0 was chosen as the representative of social-oriented modeling, with its rich constructs for capturing strategic dependencies and rationales among actors. KAOS was included as the exemplar of formal, system-oriented modeling, providing a benchmark for taxonomic precision through its rigorous goal refinement and formal semantics. Finally, MGM was selected to incorporate the critical, yet often overlooked, human-affective perspective, testing the taxonomy's ability to capture emotional goals and quality-of-experience requirements that are increasingly vital in human-centric systems. By benchmarking against this triad, we demonstrate RelnTa's applicability across the key dimensions of modern requirements engineering: social, formal, and affective.

Our analysis systematically mapped the interdependencies in our taxonomy to related constructs in these reference frameworks, with the complete results presented in Table 6. The evaluation examines the 16 interdependencies defined in the ReInTa taxonomy, systematically analyzing their coverage across three distinct categories of interdependencies: intra-type, cross-type, and multi-type.

For **intra-type dependencies**, iStar 2.0 and KAOS provide robust support for **And/Or** refinements in goal or task decomposition, while MGM only supports basic goal decomposition via **And** without explicit **Or**-refinement. Temporal dependencies like **Before** are uniquely supported by KAOS, whereas **contractual** interdependencies remain unaddressed across all frameworks. **Cooperation** among NFRs can be indirectly modeled in iStar and KAOS but is absent in MGM. **Conflict** dependencies are partially addressed: iStar's **Break** relationship differs conceptually from true conflict, KAOS explicitly supports **conflict**, and MGM handles it implicitly. Notably, **continuance** and **consequential** dependencies, critical for dynamic requirements management, are not covered by any framework, underscoring a significant gap.

In **cross-type dependencies**, **compliance** interdependency is partially modeled in iStar (via quality attributes) and KAOS (as constraints), but MGM lacks explicit support. **Exclusion** is represented in iStar through **Break** links, while KAOS and MGM omit this concept. **Contribution** dependencies are well-supported across all frameworks, though terminology varies (e.g., "Implicit emotional influence" in MGM).

In **cross-type dependencies**, **compliance** interdependency is partially modeled in iStar (via quality attributes) and KAOS (as constraints), but MGM lacks explicit support. **Exclusion** is represented in iStar through **Break** links and in MGM via negative emotional influences, while KAOS omits this concept. **Contribution** dependencies are well-supported across all frameworks, though terminology varies (e.g., "Implicit emotional influence" in MGM).

The evaluation demonstrates that while current frameworks capture subsets of interdependencies in ReInTa taxonomy, their fragmented and inconsistent approaches leave critical gaps, particularly in cross-type and multi-type relationships. This inconsistency in terminology and partial coverage highlights the need for standardization. ReInTa addresses these limitations by offering a unified, comprehensive taxonomy with clearly defined terms and broader coverage.

Table 6. Comparative evaluation of the Taxonomy with existing frameworks.

Dependency - req. classification	**iStar 2.0 Dalpiaz et al. (2016)**	**KAOS Van Lamsweerde (2001)**	**MGM Miller et al. (2015)**
And - (FR-to-FR)	AND-refinement (Goal/Task decomposition)	AND-refinement (Goal/Task decomposition)	Goal (requirements) decomposition
Or - (FR-to-FR)	OR refinement	OR-refinement (Alternative goal refinements)	-
Before - (FR-to-FR)	-	Before	-
Contractual - (FR-to-FR)	-	-	-
Cooperation - (NFR-to-NFR)	Could be modeled as mutual dependencies	Could be modeled via joint agent assignments	-
Conflict - (FR-to-FR or NFR-to-NFR)	"Break" - conceptually, not equivalent to Conflict	Conflict	Conflict can be captured but not explicitly considered
Continuance - (FR-to-FR or NFR-to-NFR)	-	-	-
Consequential - (FR-to-FR or NFR-to-NFR)	-	-	-
Compliance - (NFR-to-FR)	Could be modeled as a Quality with "Make" contributions from compliant tasks	Could be modeled as domain properties or constraints, not as a distinct dependency type.	-
Exclusion - (FR-to-NFR)	Can be represented as a "break" link	-	-
Contribution - (FR-to-NFR or NFR-to-FR)	Contribution	Contribution	Implicit emotional/quality influence
Satisfies - (FR/NFR-to-FR/NFR)	Satisfy	Operationalization	is Implicitly captured only in the context of fulfilling a goal via its sub-goals
Supports - (FR/NFR-to-FR/NFR)	Supports	Supports	-
Requires - (FR/NFR-to-FR/NFR)	NeededBy	Only in goal-to-agent assignments	-
ICost - (FR/NFR-to-FR/NFR)	Could be modeled as Quality (Cost) and contributions	-	-
CValue - (FR/NFR-to-FR/NFR)	Could be modeled as Quality (value) and contributions	Could be modeled as contribution to soft goals (e.g., "MinimizeCost")	Only emotional/quality value contribution

8 Demonstration of Applicability

Following design science principles Hevner et al. (2004); Venable et al. (2012), we evaluate the utility of the RelnTa taxonomy by applying it to the Motivational Goal Model (MGM) framework Miller et al. (2015). This practical application serves to demonstrate RelnTa's relevance and its capacity to extend and enhance existing modeling frameworks. Specifically, we utilize ReInTa to introduce new interdependencies to MGM, thereby enhancing both the robustness and expressiveness of the resulting model. This expansion enables a richer understanding of the interdependencies between functional, quality, and emotional goals, leading to a more nuanced and complete representation of system requirements. This builds on our earlier demonstration that ReInTa encompasses a broader range of requirements interdependencies than MGM (see Table 6).

To demonstrate this, we apply ReInTa to a real-world example derived from the PHArA-ON (Pilots for Healthy and Active Ageing) project[3], an H2020 European initiative aimed at developing smart and active living solutions for Europe's aging population. Figure 5 depicts the requirements model for "Support the well-being of older adults" represented in the MGM language Miller et al. (2015). The original example has been extended to incorporate the complete set of concepts from MGM, including negative emotions and several quality and emotional goals[4]. 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-functional 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. For example, the functional goal "Improve Digital Skills" is associated with "Guided", "Adaptive", "Interactive", "Easy to use", "Consideration of skills", and "Helpful", quality goals as well as "Included" emotional goal, and "Overwhelmed" as negative emotions that should be avoided. Finally, the roles responsible for achieving/activating the goals are added to the model, namely: The Older adult in our example.

As illustrated in the figure and established in the preceding analysis, the MGM framework provides only a limited set of interdependencies for capturing the different relationships among requirements. In the rest of this section, we first analyze the interdependencies supported by MGM. Subsequently, we demonstrate how ReInTa's additional interdependencies - not accounted for in MGM - can extend both the scope and depth of MGM's requirements modeling capabilities.

1. Interdependencies supported by ReInTa and MGM. ReInTa encompasses nearly all core MGM interdependencies, and we will discuss how they are covered in the following discussion.

And interdependency is explicitly captured in MGM as shown in the figure, where the top-level functional goal *Support the well-being of older adults* is refined into three lower-level functional goals via And interdependency.

[3] `https://www.pharaon.eu/`

[4] Both emotional and quality goals are widely classified as NFRs

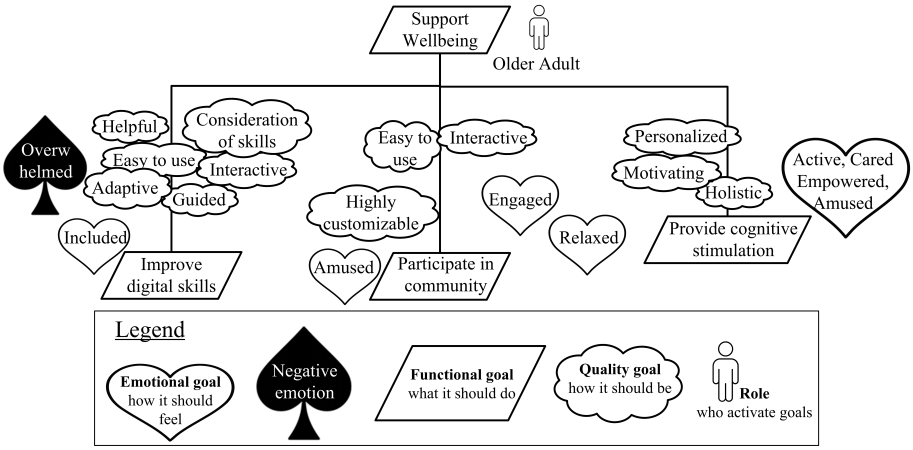

Fig. 5. MGM for Supporting the Well-being of older adults Gharib et al. (2024).

Conflict interdependency is implicitly considered in MGM, as MGM lacks a dedicated construct for such interdependency. For instance, the emotional goal *Included* is conflicting with the negative emotion *Overwhelmed*, which might result from the FR *Improve digital skills* as it requires intensive training or complex interfaces. This could trigger feelings of being overwhelmed (e.g., information overload, rapid skill demands). Another example is the conflict between *Easy to use* and *Highly customizable* as the former prioritizes simplicity, minimal steps, and intuitive interfaces, and the latter introduces complexity (e.g., settings, preferences), potentially overwhelming novice users. Note that the dotted link *Conflict* (represented in Fig. 6) does not exist in MGM since conflict is implicit, and has been added to the figure to facilitate comprehension.

Contribute interdependency can be captured implicitly in MGM as an influence of an emotional or quality goal over functional goals. For instance, both the emotional goal *Included* and the quality goal *Adaptive* contribute to the functional goal *Improve digital skills* in Fig. 6.

Satisfies interdependency In MGM, Satisfies is only captured in the context of fulfilling a goal through its sub-goals. In contrast, ReInTa models Satisfies as a bidirectional relationship between FR and NFR. For example, the functional goal *Implement gamified progress tracking* satisfies the emotional goal *Engaged*, illustrating how an FR can directly fulfill an NFR. This relationship is essential for capturing how FRs/NFRs are realized through other FRs/NFRs. Again, the dotted link *Satisfies* (shown in Fig. 6) does not exist in MGM since satisfies is not captured in this context, and there is no dedicated construct for it.

CValue interdependency can be captured implicitly in MGM as the realization of a goal/requirement can increase the value of another. For example, the quality goal *Easy to use* can increase the value of the emotional goal *Engaged* contributes to the functional goal *Improve digital skills*. Since no dedicated construct existed for this interdependency, we introduced it into the model as a dotted directed relationship, as illustrated in Fig. 6.

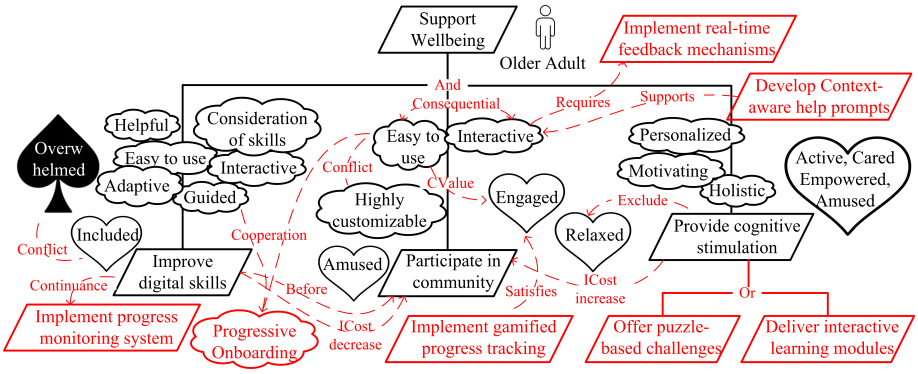

Fig. 6. Extended version of the MGM model for Supporting the Well-being of older adults

2. Interdependencies supported by ReInTa but absent in MGM. ReInTa introduces novel interdependencies not present in MGM. In the following discussion, we demonstrate how these extensions (shown in Fig. 6) enhance both the robustness and expressiveness of MGM.

Or interdependency we extend MGM with this interdependency and provide an example of its usage by refining the functional goal *Provide cognitive stimulation* via or into two sub-functional goals *Offer puzzle-based challenges* and *Deliver interactive learning modules*, i.e., the system could use puzzles OR learning modules to achieve cognitive stimulation. Additionally, to ensure consistency in representation, we enhanced the And refinement construct by explicitly including the term “And”.

Before interdependency we extend MGM with this interdependency and show how it can be used to require the realization of *Improve Digital Skills* before *Participate in the Community* as users must first acquire basic digital skills (prerequisite) to meaningfully engage in community platforms (e.g., forums, social media). E.g., a senior citizen must learn to use a tablet (Improve Digital Skills) before joining an online hobby group (Participate in the Community).

Contractual interdependency is essential for modeling systems where information flow between components must be explicitly defined. However, since MGM lacks any representation of data or information as a first-class concept, Contractual interdependency cannot be meaningfully expressed within the MGM framework.

Cooperation interdependency formalizes how synergistic interactions between requirements collectively achieve higher-level objectives. Considering it in MGM allows capturing how the quality goals *Easy to use* and *Guided* cooperate to fulfill the quality goal *Progressive Onboarding*. Here, first *Easy to use* ensures low entry barriers while *Guided* provides structured learning. Their combined effect enables seamless user acclimation by balancing accessibility with systematic instruction—precisely realizing *Progressive Onboarding* without overwhelming complexity.

Continuance interdependency adopting it in MGM will help uncover FRs that might otherwise be missed by highlighting dependencies between existing needs. For example, the functional goals *Improve digital skills* consequentially trigger *Implement*

progress monitoring system as tracking skill development becomes a necessary supporting requirement. Without this interdependency, the monitoring system might be overlooked, leading to an incomplete solution.

Consequential interdependency considering this interdependency in MGM will ensure that requirements remain logically consistent and operationally viable as they evolve throughout the requirements engineering process. For example, the quality goals *Easy to use* and *Interactive* have a consequential interdependency as adding new interactive features (e.g., drag-and-drop, live previews) requires compensatory ease-of-use measures (e.g., tooltips), while easy to use goals may constrain interaction complexity.

Compliance interdependency can proactively embeds compliance into system design when considered in MGM. For example, the functional goal *Participate in community* must comply with the quality goal *GDPR-Compliant processing* as user-generated content contains personal information and might pose a threat to users' privacy.

Exclusion interdependency considering this interdependency in MGM will pose a strict constraint on the inclusion of some NFRs that can compromise the fulfillment of FRs. For instance, the functional goal *Provide cognitive stimulation* fundamentally excludes the emotional goal *Relaxed*, as stimulating mental engagement (requiring active attention) directly contradicts the passive, stress-free state implied by relaxation. This interdependency serves as a vital design guardrail, preventing incompatible requirements from being implemented simultaneously.

Supports interdependency considering this interdependency in MGM will provide essential insights into how requirements might influence each other. For instance, the functional goal *Develop context-aware help prompts* supports the quality goal *Interactive*, as delivering real-time, adaptive help enhances interactivity.

Requires interdependency is particularly vital in complex systems where requirements have cascading dependencies. Considering it in MGM, will force explicit acknowledgment of technical and operational prerequisites during design. For example, the quality goal *Interactive* requires the functional goal *Implement real-time feedback mechanisms* since *Interactive* cannot be fulfilled without implementing real-time feedback features.

ICost interdependency considering this interdependency in MGM will capture the impact of implementing one requirement on the cost of implementing another one. For example, the functional goal *Improve Digital Skills* decreases the implementation cost of the functional goal *Participate in the Community* as users with stronger digital skills (from the first FR) require fewer moderation resources (reduced admin overhead) and simpler community interfaces, to mention a few.

This demonstration shows that RelnTa meaningfully extends MGM by introducing critical new interdependencies (e.g., Continuance, Cooperation, iCost) while fully supporting MGM's original relationships. Using the PHArA-ON case study, we illustrate how ReInTa captures subtle yet essential requirement interactions—such as timing constraints (Before), cost impacts (iCost), and mutual exclusions—that MGM cannot express. By enabling systematic analysis of trade-offs, compliance, and value delivery, ReInTa offers a practical upgrade to MGM for modeling complex systems.

9 Threats to validity

This section discusses potential threats to the validity of the study. Following Runeson et al. Runeson and Höst (2009), we categorize these threats into four types:

1. Construct validity addresses whether the study accurately measures the intended concepts and relationships Runeson and Höst (2009). A key threat in this regard is *systematic error*, which could emerge during the design or execution of our SLR. To minimize this risk, we employed multiple safeguards: First, we grounded our review protocol in established SLR methodologies, ensuring methodological rigor. Second, we maintained strict adherence to the protocol throughout all review phases, including the formulation of the search strategy, study selection, and data extraction. Third, we implemented peer verification, where two researchers independently validated key decisions to reduce subjective bias. These measures collectively enhance the reliability of our findings by aligning operational procedures with theoretical constructs.

2. Internal validity examines whether unaccounted factors may have influenced the study's outcomes, ensuring the observed effects accurately reflect the true relationships under investigation Runeson and Höst (2009). A critical threat in this regard is publication bias, a well-documented phenomenon where studies with statistically significant or positive results are more likely to be published than those with null or negative findings Kitchenham (2004). To mitigate this threat, we considered: (1) explicit inclusion of studies reporting failed or neutral results about interdependency applications (see Table 1 criteria); and (2) quality assessment using four questions (Table 2) that evaluated studies regardless of their outcome directionality. These measures help ensure our findings reflect the true spectrum of evidence in requirements interdependency research.

3. External validity examines the generalizability of our findings to broader contexts in requirements engineering Runeson and Höst (2009). A primary threat to external validity is *study completeness* - the practical impossibility of identifying every relevant study in the field. While exhaustive coverage remains theoretically unattainable, we systematically search across four major digital libraries (IEEE Xplore, ACM DL, Scopus, and Web of Science) using rigorously developed search strings; and (3) predefined inclusion/exclusion criteria (Table 1) to ensure methodological consistency. These measures collectively enhance the likelihood that our taxonomy reflects the true diversity of requirements interdependencies in both research and practice, strengthening confidence in its applicability beyond the immediate study context.

4. Reliability threats evaluate the consistency and reproducibility of our study's methodology and findings. 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://doi.org/10.5281/zenodo.16731599`

10 Conclusions and future work

This paper presented ReInTa, a comprehensive taxonomy for requirements interdependencies developed through a systematic literature review and demonstrated its applicability through practical example. The taxonomy addresses critical gaps in existing frameworks by standardizing terminology, expanding coverage to include previously unaddressed relationships, and demonstrating practical utility through its extension of the Motivational Goal Model. By organizing 16 interdependencies into three clear categories (intra-type, Cross-type, and Multi-type), ReInTa provides researchers and practitioners with a more complete framework for understanding and managing complex requirement relationships. The comparative evaluation against established frameworks (iStar 2.0, KAOS, and MGM) confirmed ReInTa's superior coverage. The application to the PHArA-ON project showcased ReInTa's ability to enhance requirements modeling, especially for systems involving emotional and quality goals. These contributions represent a significant step forward in requirements engineering, offering both theoretical foundations and practical applications for managing interdependencies throughout the software development lifecycle.

Several promising directions emerge for extending this research. First, large-scale empirical studies across different industrial domains would help validate ReInTa's generalizability and identify potential adaptations for specific contexts. Second, developing tool support to automate dependency identification and analysis could significantly enhance practical adoption. Third, the taxonomy could be extended to address emerging challenges in AI-driven systems and IoT environments, where requirements interdependencies may exhibit unique characteristics. Finally, research could explore incorporating probabilistic measures to assess dependency strength and impact, enabling more informed decision-making. These future efforts would further strengthen ReInTa's position as a comprehensive framework for requirements interdependency management.

Acknowledgments

This work was supported by the Estonian Research Council grant "Developing human-centric digital solutions" (TEM-TA120).

References

Alzyoudi, R., Almakadmeh, K., Natoureah, H. (2015). A Probability Algorithm for Requirement Selection In Component-Based Software Development, *Proceedings of the International Conference on Intelligent Information Processing, Security and Advanced Communication*, Association for Computing Machinery (ACM), pp. 1–6.

Arnold, R., Bohner, S. (1996). *Software Change Impact Analysis (Practitioners)*. `http://www.amazon.com/Software-Change-Impact-Analysis-Practitioners/dp/0818673842`

Aurum, A., Wohlin, C. (2005). Requirements engineering: Setting the context, *Engineering and Managing Software Requirements*, Springer, pp. 1–15.

Bass, L., Clements, P. C., Kazman, R. (1997). *Software Architecture in Practice*.

Berander, P. (2004). Using students as subjects in requirements prioritization, *Proceedings - 2004 International Symposium on Empirical Software Engineering, ISESE 2004*, IEEE, pp. 167–176.

Carlshamre, P., Sandahl, K., Lindvall, M., Regnell, B., Natt och Dag, J. (2001). An industrial survey of requirements interdependencies in software product release planning, *Proceedings of the IEEE International Conference on Requirements Engineering*, IEEE, pp. 84–91.

Chung, L., Nixon, B., Yu, E., Mylopoulos, J. (1999). Non-Functional Requirements in Software Engineering, *Conceptual modeling: Foundations and applications* p. 472.

Dahlstedt, Å. G., Persson, A. (2005). Requirements interdependencies: State of the art and future challenges, *Engineering and Managing Software Requirements* pp. 95–116.

Dalpiaz, F., Franch, X., Horkoff, J. (2016). iStar 2.0 Language Guide, *Technical report*. `http://arxiv.org/abs/1605.07767`

Dubois, E., Hagelstein, J., Lahou, E., Ponsaert, F., Rifaut, a. (1986). A knowledge representation language for requirements engineering, *Proceedings of the IEEE* **74**(10), 1431–1444.

Egyed, A., Grünbacher, P. (2004). Identifying requirements conflicts and cooperation: How quality attributes and automated traceability can help, *IEEE Software* **21**(6), 50–58.

Gharib, M., Falco, M., Nijboer, F., Tinga, A. M., D'Agostini, S., Rovini, E., Fiorini, L., Cavallo, F., Taveter, K. (2024). Dealing with Emotional Requirements for Software Ecosystems: Findings and Lessons Learned in the PHArA-ON Project, *in*, *Proceedings of the 18th International Conference on Research Challenges in Information Science (RCIS24)*, Cham: Springer Nature Switzerland, pp. 99–114. `https://link.springer.com/10.1007/978-3-031-59465-6_7`

Gharib, M., Giorgini, P., Mylopoulos, J. (2021). COPri v.2 — A core ontology for privacy requirements, *Data and Knowledge Engineering* **133**, 101888. `https://linkinghub.elsevier.com/retrieve/pii/S0169023X2100015X`

Gharib, M., Salnitri, M., Paja, E., Giorgini, P., Mouratidis, H., Pavlidis, M., Ruiz, J. F., Fernandez, S., Siria, A. D. (2016). Privacy Requirements: Findings and Lessons Learned in Developing a Privacy Platform, *the 24th International Requirements Engineering Conference, RE 2016*, IEEE, pp. 256–265.

Greenspan, S., Mylopoulos, J., Borgida, A. (1982). Capturing more world knowledge in the requirements specification, *Proc. 6th international conference on Software Engineering*, IEEE Computer Society Press, pp. 225–234. `http://dl.acm.org/citation.cfm?id=800254.807765`

Guan, H., Cai, G., Zhao, C. (2021). An Automatic Approach to Extracting Requirement Dependencies based on Semantic Web, *The 8th International Conference on Dependable Systems and Their Applications*, IEEE, pp. 414–420.

Guo, W., Zhang, L., Lian, X. (2021). Putting software requirements under the microscope: Automated extraction of their semantic elements, *Proceedings of the IEEE International Conference on Requirements Engineering*, pp. 416–417.

Gupta, A., Gupta, C. (2022). A novel collaborative requirement prioritization approach to handle priority vagueness and inter-relationships, *Journal of King Saud University - Computer and Information Sciences* **34**(5), 2288–2297. `https://www.sciencedirect.com/science/article/pii/S1319157819310511`

Helfert, M., Herrmann, C. (2002). Proactive Data Quality Management for Data Warehouse Systems - A Metadata based Data Quality System, *4th International Workshop on Design and Management of Data Warehouses (DMDW 2002)*, Vol. 2002, pp. 97–106. `http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS/Vol-58/herrmann.pdf`

Hevner, A. R., March, S. T., Park, J., Ram, S. (2004). Design science in information systems research, *MIS Quarterly: Management Information Systems* **28**(1), 75–105.

In, H., Boehm, B. W. (2001). Using WinWin Quality Requirements Management Tools: A Case Study, *Annals of Software Engineering* **11**(1), 141–174.
https://link.springer.com/article/10.1023/A:1012547320602

Jaeger, M. C., Hoffmann, A. (2008). Assessing Relations between Non-Functional Requirements, *Lecture Notes in Informatics (LNI), Proceedings - Series of the Gesellschaft fur Informatik (GI)*, Vol. P-122, pp. 451–456.
https://dl.gi.de/bitstreams/4e5d24b2-05c8-45f2-8872-9c2c1cfc403a/download

Karlsson, L. (2004). Requirements prioritisation: an experiment on exhaustive pair-wise comparisons versus planning game partitioning, *"8th Internation Conference on Empirical Assessment in Software Engineering (EASE 2004)" Workshop - 26th International Conference on Software Engineering* (Ease), 145–154.
http://link.aip.org/link/IEESEM/v2004/i920/p145/s1&Agg=doi

Kitchenham, B. (2004). Procedures for Performing Systematic Reviews, *Technical Report 2004*.

Kotonya, G., Sommerville, I., Kotonya, G. (1998). *Requirements engineering: processes and techniques*, 1st edn, Wiley Publishing.

Kulshreshtha, V., Boardman, J., Verma, D. (2012). The emergence of requirements networks: The case for requirements inter-dependencies, *International Journal of Computer Applications in Technology* **45**(1), 28–41.
https://www.inderscienceonline.com/doi/abs/10.1504/IJCAT.2012.050130

Leffingwell, D., Widrig, D. (2000). *Managing software requirements: a unified approach*, Addison-Wesley Professional.

Martinez, G. G., Carpio, A. F. D., Gomez, L. N. (2019). A model for detecting conflicts and dependencies in non-functional requirements using scenarios and use cases, *Proceedings 45th Latin American Computing Conference, CLEI*, pp. 1–8.
https://ieeexplore.ieee.org/abstract/document/9073957/

Miller, T., Pedell, S., Lopez-Lorca, A. A., Mendoza, A., Sterling, L., Keirnan, A. (2015). Emotion-led modelling for people-oriented requirements engineering: The case study of emergency systems, *Journal of Systems and Software* **105**, 54–71.
https://www.sciencedirect.com/science/article/pii/S0164121215000667

Monostori, L., Viharos, Z. J., Markos, S. (2000). Satisfying various requirements in different levels and stages of machining using one general ANN-based process model, *Journal of Materials Processing Technology* **107**(1-3), 228–235.

Mylopoulos, J., Amyot, D., Logrippo, L., Parvizimosaed, A., Sharifi, S. (2020). Social dependence relationships in requirements engineering, *CEUR Workshop Proceedings*, Vol. 2641, pp. 55–60.

Noviyanto, F., Razali, R., Nazree, M. Z. A. (2023). Understanding requirements dependency in requirements prioritization: a systematic literature review, *International Journal of Advances in Intelligent Informatics* **9**(2), 249.
http://ijain.org/index.php/IJAIN/article/view/1082

Nuseibeh, B., Easterbrook, S., Russo, A. (2000). Leveraging inconsistency in software development, *Computer* **33**(4), 24–29.

Robinson, W. N., Pawlowski, S. D., Volkov, V. (2003). Requirements Interaction Management, *ACM Computing Surveys* **35**(2), 132–190.

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

Sommerville, I. (2007). *Software engineering*, pearson Education limitd.

Sommerville, I., Sawyer, P. (1997). *Requirements engineering: a good practice guide*, John Wiley & Sons, Inc.

Soomro, S., Hafeez, A., Shaikh, A., Musavi, S. H. A. (2013). Ontology based requirement interdependency representation and visualization, *Communications in Computer and Information Science*, Vol. 414, Springer Verlag, pp. 259–270.
https://link.springer.com/chapter/10.1007/978-3-319-10987-9_24

Sumesh, S., Krishna, A. (2022). Sensitivity Analysis of Conflicting Goals in the i* Goal Model, *Computer Journal* **65**(6), 1434–1460.
https://academic.oup.com/comjnl/article-abstract/65/6/1434/6132358

Sumesh, S., Krishna, A., Subramanian, C. M., Murtagh, F. (2019). Game Theory-Based Reasoning of Opposing Non-functional Requirements using Inter-actor Dependencies, *Computer Journal* **62**(11), 1557–1583.
https://academic.oup.com/comjnl/article-abstract/62/11/1557/5288329

Svensson, R. B., Regnell, B., Aurum, A. (2010). Towards modeling guidelines for capturing the cost of improving software product quality in release planning, *ACM International Conference Proceeding Series*, pp. 20–23.

Tabassum, M. R., Siddik, M. S., Shoyaib, M., Khaled, S. M. (2014). Determining interdependency among non-functional requirements to reduce conflict, *International Conference on Informatics, Electronics and Vision*, IEEE, pp. 1–6.

Terry Bahill, A., Henderson, S. J. (2005). Requirements development, verification, and validation exhibited in famous failures, *Systems Engineering* **8**(1), 1–14.

Tong, Z. X., Su, X. H., Ding, X., Lin, J. X. (2018). Optimizing software development requirements based on dependency relations, *Journal of Information Hiding and Multimedia Signal Processing* **9**(4), 1014–1030.

Van Lamsweerde, A. (2001). Goal-oriented requirements engineering: A guided tour, *Requirements Engineering, 2001. Proceedings. Fifth IEEE International Symposium on*, IEEE, pp. 249–262.

van Lamsweerde, A., Darimont, R., Massonet, P. (1995). Goal-directed elaboration of requirements for a meeting scheduler: problems and lessons learnt, *Proceedings of the IEEE International Conference on Requirements Engineering*, Published by the IEEE Computer Society, pp. 194–203.

Vasilache, S., Tanaka, J. (2005). Bridging the gap between analysis and design using dependency diagrams, *Third ACIS International Conference on Software Engineering Research, Management and Applications, SERA 2005*, Vol. 2005, pp. 407–414.
https://ieeexplore.ieee.org/abstract/document/1563190/

Venable, J., Pries-Heje, J., Baskerville, R. (2012). A comprehensive framework for evaluation in design science research, *International Conference on Design Science Research in Information Systems*, Springer, pp. 423–438.

Wiegers, K., Beatty, J. (2013). *Software requirements*.

Yu, E. S. K. (1993). Modeling organizations for information systems requirements engineering, *Requirements Engineering, 1993., Proceedings of IEEE International Symposium on*, IEEE, pp. 34–41.

Zave, P., Jackson, M. (1997). Four dark corners of requirements engineering, *ACM Transactions on Software Engineering and Methodology (TOSEM)* **6**(1), 1–30.

Received August 3, 2025 , revised September 30, 2025, accepted November 5, 2025