Journal of Software Engineering Research and Development, 2019, 7:9, doi: 10.5753/jserd.2019.458

A revisited systematic literature mapping on the support of requirement patterns for the software development life cycle

Taciana N. Kudo [**DC-UFSCar, São Carlos-SP, Brazil** | ***taciana@dc.ufscar.br***]
Renato F. Bulcão-Neto [**INF-UFG, Goiânia-GO, Brazil** | ***rbulcao@ufg.br***]
Alessandra A. Macedo [**FFCLRP-USP, Ribeirão Preto-SP, Brazil** | ***ale.alaniz@usp.br***]
Auri M. R. Vincenzi [**DC-UFSCar, São Carlos-SP, Brazil** | ***auri@dc.ufscar.br***]

Abstract In the past few years, the literature has shown that the practice of reuse through requirement patterns is an effective alternative to address specification quality issues, with the additional benefit of time savings. Due to the interactions between requirements engineering and other phases of the software development life cycle (SDLC), these benefits may extend to the entire development process. This paper describes a revisited systematic literature mapping (SLM) that identifies and analyzes research that demonstrates those benefits from the use of requirement patterns for software design, construction, testing, and maintenance. In this extended version, the SLM protocol includes automatic search over two additional search sources, the application of the snowballing technique, and the quality assessment of the relevant ten-study-group for data analysis and synthesis. In comparison to previous work, results still show a small number of studies on requirement patterns at the SDLC (excluding requirements engineering). Results indicate that there is yet an open field for research that demonstrates, through empirical evaluation and usage in practice, the pertinence of requirement patterns at software design, construction, testing, and maintenance.

Keywords: *Requirement pattern, Software development life cycle, Systematic literature mapping*

1 Introduction

Requirements engineering is a critical development phase in which software functionalities and constraints must be well identified and understood. However, a high percentage of software projects do not meet deadlines and budget due to incomplete, misinterpreted, conflicting, or omitted requirements (Tockey, 2015; Palomares et al., 2017).

To deal with this issue of quality of requirements specifications, software requirement patterns (SRP) have been given special attention in the recent years (Palomares et al., 2017; Irshad et al., 2018). An SRP is an abstraction that groups both behaviors and services of applications with similar characteristics. It works as a template for new requirements specification, and it can also be replicated in future requirements documentation (Withall, 2007). For instance, to write a user authentication functional requirement, one can use an SRP for this purpose and make appropriate adaptations to the requirement, if necessary.

Several proposals for SRPs are found in the literature such as for embedded (Konrad and Cheng, 2002), content management (Palomares et al., 2013), and cloud computing systems (Beckers et al., 2014). Among the benefits obtained with the adoption of SRPs are: (i) greater efficiency in requirements elicitation since these are not identified from scratch; (ii) quality and consistency improvement in the requirements specification document; and (iii) improved requirements management (Withall, 2007).

Because of the inherent interaction between requirements engineering and other phases of the software development life cycle (SDLC), it is assumed that the benefits of using SRPs can reach other development activities. Although there are secondary studies on software engineering (Kitchenham and Brereton, 2013), requirements engineering (Curcio et al., 2018), and requirement patterns (Barros-Justo et al., 2018), there is no evidence of secondary studies that analyze the use of SRPs at other SDLC phases. In short, existing secondary studies are restricted to analyzing the adoption of SRPs exclusively in the requirements engineering phase.

In recent work, we performed a systematic literature mapping (SLM) that identifies and analyses primary studies that put in evidence the usage of SRPs at the software design, construction, testing, and maintenance[1] phases (Kudo et al., 2019a). The underlying protocol included an automatic search over four sources of information and the definition and application of inclusion and exclusion criteria over 117 non-duplicate studies found. Only nine primary studies were considered relevant, given the research aim (Kudo et al., 2019a).

Results indicated that most of the relevant studies apply SRPs in software design, but none in software maintenance. Moreover, only one study was featured as validation research, while the remaining studies were solution proposals. Thus, we concluded that the benefits from the SRPs usage in practice at other SDLC phases are still in its early stages.

In this paper, we revisit the SLM described in Kudo et al. (2019a) and improve the identification and selection methods of primary studies. Besides the inclusion of two additional sources of information in the automatic search process, we also perform the snowballing technique (Wohlin, 2014) that identifies relevant studies through the scanning of the list of bibliographic references or citations of a paper.

The inclusion of two sources of studies resulted in 32 extra, non-duplicate papers, from which one novel relevant study arose. Considering the 9 relevant primary studies found in our previous work, we obtained a ten-primary-study group in this research. To check whether other essential studies on

[1] We adopt the terminology of the Software Engineering Body of Knowledge (SWEBOK) for the SDLC phases (Bourque and Fairley, 2014).

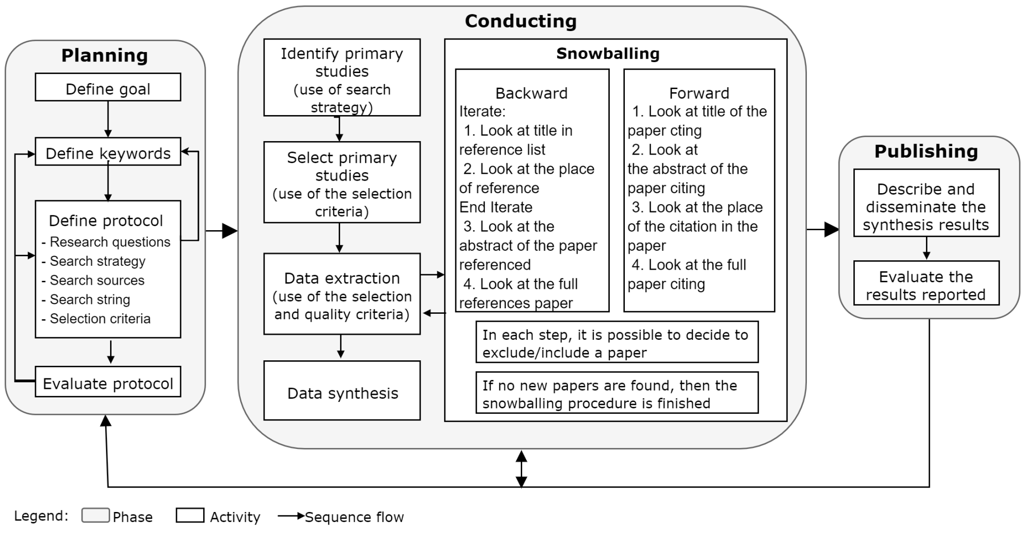
Planning
Define goal
Define keywords
Define protocol
- Research questions
- Search strategy
- Search sources
- Search string
- Selection criteria
Evaluate protocol
Conducting
Identify primary studies (use of search strategy)
Select primary studies (use of the selection criteria)
Data extraction (use of the selection and quality criteria)
Data synthesis
Snowballing
Backward
Iterate:
1. Look at title in reference list
2. Look at the place of reference
End Iterate
3. Look at the abstract of the paper referenced
4. Look at the full references paper
Forward
1. Look at title of the paper cting
2. Look at the abstract of the paper citing
3. Look at the place of the citation in the paper
4. Look at the full paper citing
In each step, it is possible to decide to exclude/include a paper
If no new papers are found, then the snowballing procedure is finished
Publishing
Describe and disseminate the synthesis results
Evaluate the results reported
Legend: Phase Activity Sequence flow

Figure 1. Phases and activities of this SLM, adapted from (Fabbri et al., 2013; Wohlin, 2014)

this research exist, we also analyzed the list of bibliographic references as well as the citing papers of each one of these 10 studies. The snowballing technique resulted in 202 non-duplicate papers from which none was assessed as relevant after the re-application of inclusion and exclusion criteria.

We read the full text of 10 studies to extract the answers to the SLM research questions and, in parallel, to assess the quality of each relevant paper. Finally, we synthesized in a bubble graph a map with the remarkable characteristics of this ten-study-group. In comparison with our previous work, results continue to point out a lack of research on SRPs for software design, construction, testing, and maintenance.

The organization of this paper is as follows. Section 2 details the protocol of this SLM. Section 3 reports the data extraction and the quality assessment activities regarding the relevant studies. The answers to the research questions in this study and the research gaps are summarized in Section 4. Finally, Section 5 describes the validation threats of this SLM, whereas Section 6 presents our final remarks.

2 The Systematic Mapping Protocol

In general, a systematic study process can be divided into three distinct phases (Fabbri et al., 2013): planning, conduction, and publishing of results. First, a protocol is planned in such a way one can reproduce it later. This systematic mapping protocol includes the definition of the main goal, research questions, search strategy, search string, sources of studies, and inclusion and exclusion criteria.

In the conduction phase, studies gathered from search engines and bibliographic databases are identified and selected using the inclusion and exclusion criteria previously defined. A set of useful information is extracted from these selected studies that, in turn, can be still excluded from the SLM. Snowballing is performed over these included papers by firstly checking their references list. The selection of the studies from this backward analysis is also based on the previous reading of the paper's title and abstract. This same process is also carried out with the citation list of the same papers examined in the data extraction step. Forward and backward analyses finish when no new study is included. Following the SLM goal, the studies remaining constitute the set of relevant papers from which answers for the research questions of the protocol are analyzed and synthesized. A quality assessment activity is also conducted to assist data synthesis from these relevant papers, as suggested by Kitchenham et al. (2010).

In the publishing phase, the entire protocol and the results of each previous stage are documented as scientific papers or technical reports. The SLM presented in this paper is an extension of the Kudo et al. (2019a)'s work and follows those three phases, as depicted in Figure 1.

2.1 Research questions and keywords

The main goal of this SLM is to identify studies that explore the benefits of requirement patterns for every SDLC phase, except for the requirements engineering process. Based on this goal, the set of research questions (**RQ**) this SLM should answer, and the respect justifications are presented next:

RQ1. *At what SDLC phases are requirement patterns used: design, construction, testing, and/or maintenance?* This question is essential to find out if there is research on requirement patterns covering other SDLC phases, beyond requirements engineering.

RQ2. *Is there evidence of requirement patterns usage in practice at those SDLC phases?* This question is relevant to discover empirical evidence on requirement patterns usage at other SDLC phases, i.e., not only solution proposals.

RQ3. *Are there reported benefits of using requirement patterns at those phases? If so, what metrics are used to measure these benefits?* This question is useful to find out if the benefits of requirement patterns (e.g., development time savings, better quality specifications, etc.) have been exploited at other SDLC phases. If so, we want to know how these benefits have been measured.

To support the definition of standardized terms in Software Engineering, the search terms are borrowed from the SE-VOCAB (Software and Systems Engineering Vocabulary), which is an ISO/IEEE initiative to standardize the terms used in Software Engineering (ISO/IEC/IEEE, 2017). The following is the set of keywords used for the definition of the search string: *requirement pattern*, *development process*, *software development*, *life cycle*, *design*, *construction*, *coding*, *implementation*, *test*, *integration*, and *maintenance*.

A search strategy should find relevant studies to answer the research questions. Next, we present the search strategy performed in this SLM that includes automatic search and the snowballing technique.

2.2 Automatic search

After evaluating the trade-off between coverage and relevance of the search results in a pilot search, we opted for the following combination of keywords[2] as search string:

("*requirement pattern*" OR "*requirement patterns*" OR "*requirements pattern*" OR "*requirements patterns*")
AND (("*software development*" OR "*development process*")
OR
("*life cycle*" OR *design* OR *construction* OR *coding* OR *implementation* OR *test* OR *integration* OR *maintenance*))

Besides *ACM DL*[3], *Engineering Village*, *IEEE Xplorer*, and *Scopus*, we also performed searches at the *ScienceDirect* and the *Web of Science* websites. Similarly, we did searches based on studies metadata, at least over the abstracts because of their richer content.

Table 1 describes in detail the number of studies returned per source of studies, both in the original search[4] (Kudo et al., 2019a) and in this revisited version[5]. Therefore, 85 studies were identified (including duplicate papers) after the inclusion of two new bibliographic databases (*ScienceDirect* and *Web of Science*) and the update of search results over the four original sources of studies.

Table 1. Number of studies returned per source.

Source	Original	Extension	Difference
ACM DL	24	26	2
Engineering Village	100	106	6
IEEE Xplorer	23	25	2
Scopus	71	76	5
ScienceDirect	-	9	9
Web of Science	-	61	61
Total	**218**	**303**	**85**

2.3 Selection of primary studies

This section describes the selection method of relevant studies to answer the research questions of this SLM. The same original selection criteria were applied to the 303 papers returned by the automatic search process. The exclusion criteria (EC) are:

EC1 - It is not a primary study.
EC2 - It is not a paper (e.g., preface or summary of journals or conference proceedings).
EC3 - The research is not about SRP.
EC4 - The research addresses SRP in requirements engineering only.
EC5 - The full study text is not in English.
EC6 - The full study text is not accessible.
EC7 - It is a preliminary or short version of another study.

A paper is removed from this SLM whenever it meets at least one of the exclusion criteria (EC) presented; otherwise, the study is categorized based on the following inclusion criteria (IC):

IC1 - It addresses SRP in software design.
IC2 - It addresses SRP in software construction.
IC3 - It addresses SRP in software testing.
IC4 - It addresses SRP in software maintenance.

Figure 2 depicts the entire selection process with the respective number of primary studies chosen and removed in each activity of the conduction phase.

After the automatic search process, 155 duplicate papers are identified and removed (from the 303 studies group) with the support of the StArt tool (Fabbri et al., 2016). Next, we proceeded with reading of the title, summary, and keywords of each of the 148, upon which we applied the exclusion and inclusion criteria. As a result, we selected 41 possibly relevant studies because this selection relies on the reading and interpretation of papers' metadata only.

In the data extraction activity, we read the full text of these 41 studies from which we excluded 31 papers by the EC4 criterion, i.e., their research focus is on SRP in the requirements engineering phase. We describe the process of data extraction of the 10 studies remaining in Section 3. These studies are identified throughout this paper as S1 to S10 as follows:

S1 - Adaptive requirement-driven architecture for integrated healthcare systems (Yang et al., 2010)
S2 - Analysing security requirements patterns based on problems decomposition and composition (Wen et al., 2011)
S3 - An architectural framework of the integrated transportation information service system (Chang and Gan, 2009)
S4 - Application of ontologies in identifying requirements patterns in use cases (Couto et al., 2014)
S5 - Effective security impact analysis with patterns for software enhancement (Okubo et al., 2011)
S6 - From requirement to design patterns for ubiquitous computing applications (Knote et al., 2016)
S7 - Modeling design patterns with description logics: A case study (Asnar et al., 2011)
S8 - Mutation patterns for temporal requirements of reactive systems (Trakhtenbrot, 2017)
S9 - SACS: A pattern language for Safe Adaptive Control Software (Hauge and Stølen, 2011)
S10 - Re-engineering legacy Web applications into RIAs by aligning modernization requirements, patterns and RIA features (Conejero et al., 2013)

[2]Plural variations of the term "requirement pattern" are necessary due to the capabilities of the search engines of each source of studies.
[3]We chose the *The ACM Guide to Computing Literature* because it is a most comprehensive bibliographic database on Computing, including the full-text collection of all ACM publications.
[4]Search carried out from April 24 to May 5, 2018.
[5]Additional search performed on June 3 and 4, 2019.

Table 2. The total number of studies removed per exclusion criteria throughout the conducting phase.

Activity	EC1	EC2	EC3	EC4	EC5	EC6	EC7	Total
Automatic search	7	13	40	47	0	0	0	**107**
Data extraction	0	1	10	15	0	2	3	**31**
Snowballing	2	0	186	14	0	0	0	**202**
Total	**9**	**14**	**236**	**76**	**0**	**2**	**3**	**340**

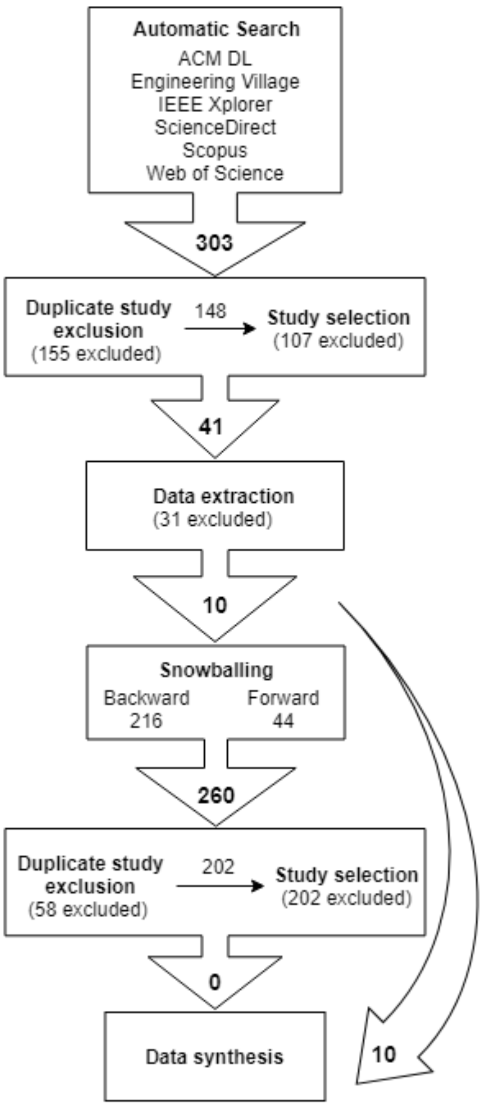

Figure 2. A detailed view of the conduction phase: automatic search, duplicate study exclusion, study selection, data extraction, snowballing, and data synthesis.

2.4 Snowballing

Besides automatic search, our search strategy includes snowballing as an attempt to obtain other relevant studies using the papers S1 to S10 as input.

Regarding backward snowballing, we collected the reference list of each paper from the Scopus database, resulting in 216 documents whose metadata (title, abstract, and keywords) we stored into the StArt tool. After the removal of 49 duplicate studies, we read the metadata of the 167 documents remaining to decide for the exclusion or the tentative inclusion of a paper for further analysis. As no new paper was found in the first round of backward snowballing, we finished this analysis earlier.

In sequence, we searched the citation list of S1 to S10 from the Scopus website, resulting in 44 papers also registered into the StArt tool. Similarly, no new paper was retrieved in the first round of this forward snowballing step, resulting from the removal of 9 duplicate studies and the reading of the metadata of the 35 documents remaining.

Both snowballing procedures end up the process of selection of relevant studies of this SLM. Figure 2 depicts the total number of studies identified (260), excluded (58), and selected (0) from the overall snowballing process. As a result, the data extraction and synthesis activities include only the studies S1 to S10 previously presented.

Finally, Table 2 summarizes the removal process of studies in the conduction phase. Most of the papers removed in the automatic search (87 of 107) are due to the EC3 and EC4 criteria, i.e., they do not address SRP, or they do it in the requirements engineering phase only, respectively. Studies were excluded at a similar rate (25 of 31) in data extraction activity. These exclusion rates around 80% are expected because of the trade-off analysis between coverage and relevance of the search string.

Differently, most of the studies removed during both snowballing procedures (186 of 202) are because of the EC3 criterion. Two related reasons explain this 92% exclusion rate: first, in general, the size of the reference list of a paper is far more extensive than the number of studies citing that paper; second, the papers in a reference list often address other research topics. Besides, only 7% of the studies referenced by or citing them represent research on SRPs (14 of 202). Even so, none of these explores SRPs at other stages of SDLC other than requirements engineering.

3 Data Extraction

This section describes the data extraction process from the full-text-reading of the 10 relevant studies (S1 to S10) of this SLM. Besides presenting a comparative analysis of the contribution types of each paper, we also extract:

1. the quality score of each primary study;
2. the type of research carried out;
3. the type of requirement addressed by SRP;
4. the SDLC phase supported by SRP;
5. and the contribution type.

3.1 Quality assessment

The quality assessment may be useful for an SLM to assure that sufficient information is available to be extracted. However, we concur with Petersen et al. (2015) that quality assessment should not pose high requirements on the primary studies because the main objective of an SLM is to give a broad overview of a research topic. Rather than using quality criteria for the exclusion of papers, our quality assessment approach assists data analysis and synthesis, such as to investigate whether different quality scores are associated with varying outcomes of the primary studies (Kitchenham et al., 2010; Petersen et al., 2015).

Multiple checklists are available in the literature to help the process of assessing the quality of primary studies. Here, we evaluated the quality of primary studies through nine quality criteria from which six are general factors (G1 to G6), as described in Jamshidi et al. (2013), and three are particular factors (P1 to P3) that we defined based on the subject of this SLM. Following is the full description of every general and specific quality criteria, including their respective predefined responses and scores (in parenthesis). Observe that G2 is the only criterion whose score ranges from 0 to 1, indicating a lower weight in the quality score of each study.

G1- Problem definition of the study.
- **(2)** : There is an explicit problem description.
- **(1)** : There is a general problem description.
- **(0)** : There is no problem description.

G2- Environment in which the study is carried out.
- **(1)** : There is an explicit description of the environment in which the research is performed (e.g., lab setting, as part of a project, in collaboration with industry, etc.).
- **(0.5)** : There are some general words about the environment in which the research is performed.
- **(0)** : There is no description of the environment.

G3- Research design of the study.
- **(2)** : There is an explicit description of the plan (different steps, timing, etc.) used to perform the research, or of the way the research is organized.
- **(1)** : There are some general words about the research plan or the way the research is organized.
- **(0)** : There is no description of the research design.

G4- Contributions of the study.
- **(2)** : There is an explicit list of the contributions/results.
- **(1)** : There are some general words about the study results.
- **(0)** : There is no description of the study results.

G5- Insights derived from the study.
- **(2)** : There is an explicit list of insights/lessons learned from the study.
- **(1)** : There are general words about insights/lessons learned from the study.
- **(0)** : There is no description of the insights derived from the study.

G6- Limitations of the study.
- **(2)** : There is an explicit list of the limitations of the study.
- **(1)** : There are general words about the limitations of the study.
- **(0)** : There is no description of the limitations of the study.

P1- The SRP structure.
- **(2)** : There is an explicit description of the SRP structure.
- **(1)** : There is some general information about the SRP structure.
- **(0)** : There is no description of the SRP structure.

P2- The integrated use of SRPs with the SDLC phases.
- **(2)** : There is an explicit description of which SDLC phase benefits from SRPs usage.
- **(1)** : There are some general words about which SDLC phase benefits from SRPs usage.
- **(0)** : There is no description of which SDLC phase benefits from SRPs usage.

P3- Empirical investigation of SRPs usage in the SDLC phases.
- **(2)** : There is an explicit description of empirical investigation.
- **(1)** : There is some general information about the empirical investigation.
- **(0)** : There is no description of empirical investigation.

The relevance of the particular quality criteria (P1 to P3) is presented next. As stated by Franch et al. (2010), the reuse of an SRP heavily depends on a detailed description of its structure (P1). The P2 criterion is important to identify the adherence of each study to the research question RQ1, i.e., the SDLC phase supported by SRPs. Finally, the P3 criterion allows distinguishing studies with empirical evidence.

Once presented general and particular quality criteria, the following is the final quality score (QS) formula that provides us with a numerical quantification as a means of ranking the relevant primary studies:

$$QS = [(\frac{\sum_{G=1}^{6}}{11}) + (\frac{\sum_{P=1}^{3}}{6} \times 3)] \tag{1}$$

, where the sums of G1 to G6 and of P1 to P3 may reach a maximum score of 1 and 3, respectively. That is, specific quality criteria represent 75% weightage in the final quality score because of their higher importance in comparison with the general items.

Table 3 presents the full quality assessment of the ten primary studies, in descending order of the final quality score (QS at the rightmost column). The respective values assigned to the general and particular quality criteria of every primary study are also available in Table 3 as well as the particular total score for general and particular quality criteria (SGC and SPC, respectively).

Observe that the P3 criterion contributes to a subclassification of the ten-study-group: research with no empirical investigation (S1 to S6, S8, and S9) got a quality score less than 3.0, while the studies whose quality score is higher than 3.0 (S7 and S10) have more empirical evidence and explicit their lessons learned (G5 criterion). However, S7 and S10 obtained a minor grade for the P1 criterion because they do not describe the structure of their SRPs proposals. Finally, the lower quality scores are mainly due to the grades of the P2 and P3 criteria. Consider the case of the studies S3 and S4 that both have no empirical evidence and partially describe how to employ SRPs in an SDLC phase.

Table 3. A detailed view of the quality assessment results.

Study	G1	G2	G3	G4	G5	G6	**SGC**	P1	P2	P3	**SPC**	**QS**
S10	2	1	2	2	2	1	0.9	1	2	2	0.8	**3.3**
S7	2	1	2	2	2	0	0.8	1	2	2	0.8	**3.2**
S6	2	1	2	2	1	0	0.7	2	2	0	0.7	**2.8**
S2	1	1	2	2	0	2	0.7	2	2	0	0.7	**2.8**
S9	2	1	2	2	1	0	0.7	2	2	0	0.7	**2.8**
S5	2	0.5	2	2	0	0	0.6	2	2	0	0.7	**2.7**
S8	2	0	2	2	0	0	0.5	2	2	0	0.7	**2.6**
S1	2	0	2	1	0	0	0.5	2	2	0	0.7	**2.6**
S4	1	0.5	2	2	0	1	0.6	2	1	0	0.5	**2.1**
S3	2	0	2	1	0	0	0.5	2	1	0	0.5	**2.0**

3.2 Research type

We classified the ten-study-group using Petersen et al. (2015)'s criteria in which a set of conditions determine the type of research developed. For instance, opinion research solely reports the author's point of view about a subject. In this case, there is no usage in practice, empirical evaluation, author's experience report, or proposal of a conceptual framework or a novel solution.

Table 4 shows that, according to Petersen et al. (2015)'s taxonomy, most of the studies (8 of 10) is a solution proposal because there is no empirical evaluation: three studies are validated by a free proof of concept, whereas the five remaining do not even confirm their proposals. Furthermore, only two of ten studies are validation research: S7 presents a case study, and S10 describes an experiment with controlled conditions.

Table 4. Types of research and validation of relevant studies.

Type of research	**Type of validation**
Solution proposal	Proof of concept: S2 S5 S9
	No validation: S1 S3 S4 S6 S8
Validation research	Case study: S7
	Experiment: S10

3.3 Type of software requirement

Next, we analyzed the particular type of software requirement covered by SRP, as presented in Table 5. Four of the relevant studies define SRP for the adaptability requirement and the other four papers for the security one. The proposals of SRP described in the remaining two studies do not address a specific type of software requirement.

Table 5. Type of requirement covered by an SRP.

Type of requirement	**Studies**
Adaptability	S1 S3 S6 S8
Security	S2 S5 S7 S9
General purpose	S4 S10

3.4 A comparative analysis

Next, we describe a detailed comparative analysis of the contributions proposed in S1 to S10, from which we perceived some similarities and identified the SDLC phase supported by their SRPs solutions.

Studies S1 and S3 propose a similar conceptual architecture for systems developed from SRPs, as illustrated in Figure 3. The dashed lines A, B, C, and D show the similarities between the architectures proposed in S1 (left-hand side) and S3 (right-hand side). The requirements layer (A) identifies, analyzes, and models requirements as user requirement patterns (URP). The service layer (B) interacts with the requirements layer and provides services to satisfy the URP. The security and information sharing mechanism (C) establishes a process of reliable information exchange between systems of the same domain. The knowledge base (D) combines standards, norms, and ontologies of the system domain. The motivation of both research efforts is the need to share information between systems of the same area: medical systems (in S1) and transport systems (in S3).

Regarding S1 and S3 again, these studies make use of SRP to support the software design phase. In both studies, a URP in the requirements layer leads to the efficient selection of services in the service layers. A URP is a crucial element not only because it represents user requirements but also due to the fact it guides the operation of the entire system.

We also observed commonalities on how S2 and S5 represent security requirements as an SRP, as depicted in Figure 4. Both studies specify security requirement patterns with similar structure and security concepts (context, assets, and threats) as well as protection measures as design patterns.

Illustrated as dashed lines in Figure 4, the steps outlined in S2 (left-hand side) — the identification of stakeholders and objectives, essential information assets, and threat sources using standards — match with the following items of the security requirement pattern in S5 (right-hand side), respectively: the pattern definition format (context, problem, solution, and structure), asset, and threat. Finally, the step "adding protection measures in the system design" in S2 matches with the countermeasure concept described as security design patterns in S5.

From this analysis, we concluded that S2 and S5 also make use of SRP to benefit the software design phase because they define security requirement patterns and relate them to design-pattern-based protection measures.

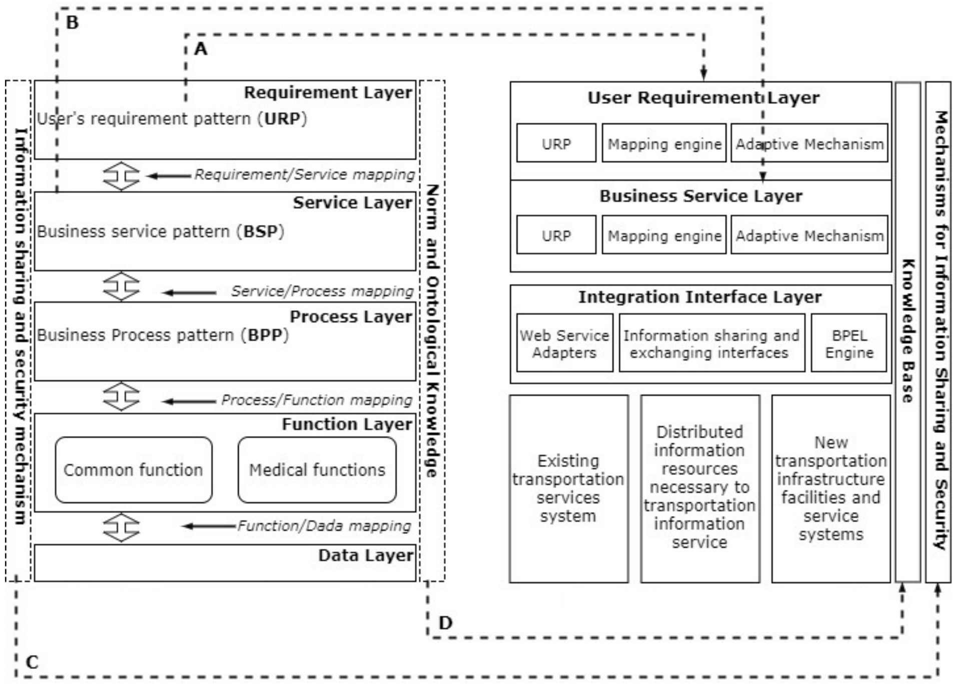
B
A
Requirement Layer
User's requirement pattern (URP)
Requirement/Service mapping
Service Layer
Business service pattern (BSP)
Service/Process mapping
Process Layer
Business Process pattern (BPP)
Process/Function mapping
Function Layer
Common function
Medical functions
Function/Dada mapping
Data Layer
Information sharing and security mechanism
Norm and Ontological Knowledge
User Requirement Layer
URP
Mapping engine
Adaptive Mechanism
Business Service Layer
URP
Mapping engine
Adaptive Mechanism
Integration Interface Layer
Web Service Adapters
Information sharing and exchanging interfaces
BPEL Engine
Existing transportation services system
Distributed information resources necessary to transportation information service
New transportation infrastructure facilities and service systems
Knowledge Base
Mechanisms for Information Sharing and Security
D
C

Figure 3. A comparative analysis of the SRP-based conceptual architectures discussed in S1 (left-hand side) and S3 (right-hand side).

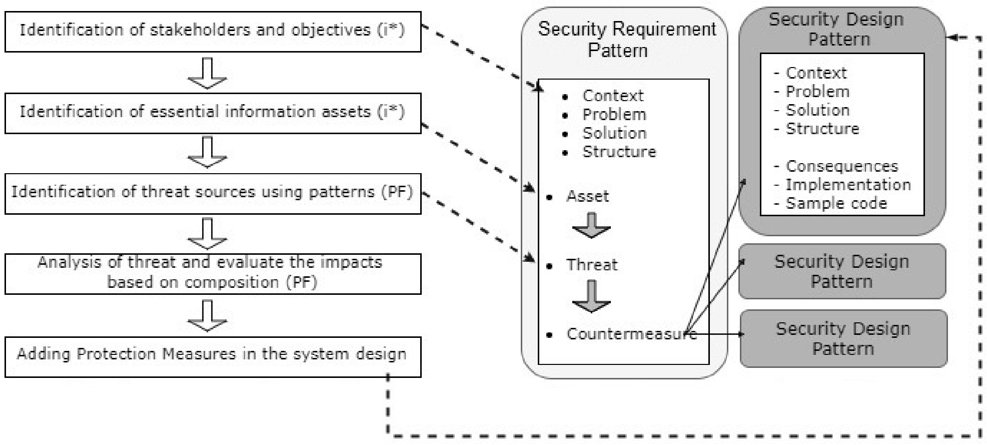
Identification of stakeholders and objectives (i*)
Identification of essential information assets (i*)
Identification of threat sources using patterns (PF)
Analysis of threat and evaluate the impacts based on composition (PF)
Adding Protection Measures in the system design
Security Requirement Pattern
Context
Problem
Solution
Structure
Asset
Threat
Countermeasure
Security Design Pattern
- Context
- Problem
- Solution
- Structure
- Consequences
- Implementation
- Sample code
Security Design Pattern
Security Design Pattern

Figure 4. A comparative analysis of the SRP-based security approaches discussed in S2 (left-hand side) and S5 (right-hand side).

As a result of the analysis of S8 and S9, we identified that both studies present proposals of requirement patterns representation format. In S8, each natural language-written requirement binds to a linear-temporal-language-written formula, in which mutations soften the likely issues in this association. Each type of requirement pattern attaches its potential failures and the respective appropriate variations. The formulas associated with mutants have multiple purposes, such as test generation, the adequacy of test sets, or the automatic construction of monitors for the system's behavior verification at run-time. Thus, the mutations included in the transformation of the requirement patterns contribute to the software testing phase.

In the case of S9, a composite pattern integrates three types of software patterns (i.e., requirement, design, and security). Based on the problem frames theory, this composite pattern uses parameters extracted from an inner requirement pattern, from which a set of functions correspond both to solutions in a design pattern and contextual elements in a security pattern. Thus, this applicability of SRP is at software design.

The studies S4 and S7 model requirement patterns using ontologies based on formal description logic. As ontology-based SRPs allow the automatic generation of source code in S4, this SRP contribution is to the software construction phase. In study S7, authors implement a mechanism that automatically binds an ontology-based security requirement pattern to a corresponding design pattern solution. Thus, the SRP's main contribution in S7 is for the design phase of the SDLC.

In the context of ubiquitous computing (ubicomp) applications, S6 aims to map dependencies between design patterns and requirement patterns. This software pattern-integration approach bridges the gaps of the early software development phase, where recurring requirements demand similar design solutions, such as the case of the adaptability requirement for ubicomp applications. Consequently, the main contribution of S6 is for the software design phase.

Regarding the study S10, it presents a systematic process to modernize legacy Web applications into Rich Internet Applications (RIA). The core of that process is a set of trace-

Table 6. Data extraction from the 10 relevant studies.

Type of contribution	SDLC phase	Type of requirement	Studies	QS
Conceptual architectures for SRP-based systems	Design	Adaptability	S1 S3	2.6 2.0
Representation formats for SRP	Design	Security	S2 S5 S9	2.8 2.7 2.8
	Testing	Adaptability	S8	2.6
Processes for discovery and use of SRP	Design	Security	S7	3.2
	Design	General purpose	S10	3.3
	Construction	General purpose	S4	2.1
Catalog of SRP	Design	Adaptability	S6	2.8

ability matrices that relate modernization requirements, RIA features, and patterns. A final traceability matrix suggests the most suitable RIA patterns for each new requirement based on the values of two different metrics: the degree of requirement realization (DRR) and the degree of pattern realization (DPR). Once selected, the RIA patterns are weaved into the legacy models so that those pattern-based RIA functionalities are incorporated into the system. The reusability of RIA patterns is very clear because the patterns traceability matrix is built once and used in any modernization process that, in turn, takes a lesser design time. Thus, in this approach, SRPs cover the gap between requirements elicitation and architectural design along the RIA development process.

3.5 Summary

Table 6 summarizes the analysis of the ten-study-group by the types of contributions identified: conceptual architectures for SRP-based systems, processes for discovery and use, representation formats, and catalogs of SRP. The final quality scores (QS) of each study are at the rightmost column.

4 Data Synthesis

This section presents a synthesis of the data extracted from the relevant studies to answer the research questions.

4.1 About the research question 1

To answer the research question "*At what SDLC phases are requirement patterns used: design, construction, testing and/or maintenance?*", eight studies use SRPs at the design phase, one at construction, one at software testing, and none at software maintenance.

Among the eight studies that address SRPs at software design (S1 to S3, S5 to S7, S9, and S10), there are no repeating authors, neither the convergence of studies to one or more research groups. Two hypotheses can explain the high concentration of studies related to the design phase: the fact that it is after requirements engineering as well as the increasing usage of design patterns in software development.

Even though the studies S3 and S4 do not clearly state the SDLC phase supported by SRPs, we consider that their SRPs proposals bring benefits to the software design and construction phases, respectively (see P2 criterion).

A significant difference between the number of relevant studies (10) and the number of papers excluded (77) is because these investigate SRPs exclusively for requirements engineering. This unbalance makes it clear that there is still an open field for research on the benefits of SRPs for the other SDLC phases, such as testing and (1) maintenance (0).

As a consequence, another evidence is the lack of research on the use of SRPs along the entire SDLC, from requirements engineering to software maintenance. An example of a challenging study could be the evaluation of the improvements for the SDLC resulting from the adoption of SRPs, beyond the well-known benefits of time savings and better quality specifications.

4.2 About the research question 2

Regarding the research question "*Is there evidence of requirement patterns usage in practice at those SDLC phases?*", there is no study that reports evidence of SRPs usage in the software industry. Eight of the ten-relevant-studies are solution proposals with no validation, and only two papers (S7 and S10) are validation research with the highest quality scores, according to our quality assessment. This analysis suggests that future work should be more focused on the use of SRPs along the SDLC in the software industry.

4.3 About the research question 3

To answer the research question "*Are there reported benefits of using requirement patterns at those phases? If so, what metrics are used to measure these benefits?*", S10 is the only study that defines SRP-related metrics. We believe that this lack of concern with metrics is because most articles are solution proposals, thus without use in practice.

In S10, the metrics DRR (degree of requirement realization) and DPR (degree of pattern realization) select candidate RIA patterns in the process of re-engineering of legacy web applications. A value of 1 in DRR indicates that a pattern fully supports all the RIA features demanded by the requirement, whereas a value of 0 means that the requirement and the pattern do not share any feature. Similarly, a value of 1 in DPR denotes that the requirement demands all the RIA features supported by the pattern. In contrast, a value close to 0 implies that the requirement needs an insignificant amount of the RIA features supported by the pattern.

The experiment results in S10 show that, in the worst case, more than half of the patterns would have been automatically suggested by the authors' method. Furthermore, the synchronization patterns indicated by the approach and those used by developers are the same in all systems tested in the experiment. Both results allow concluding that SRPs usage in S10 implies significant development time savings.

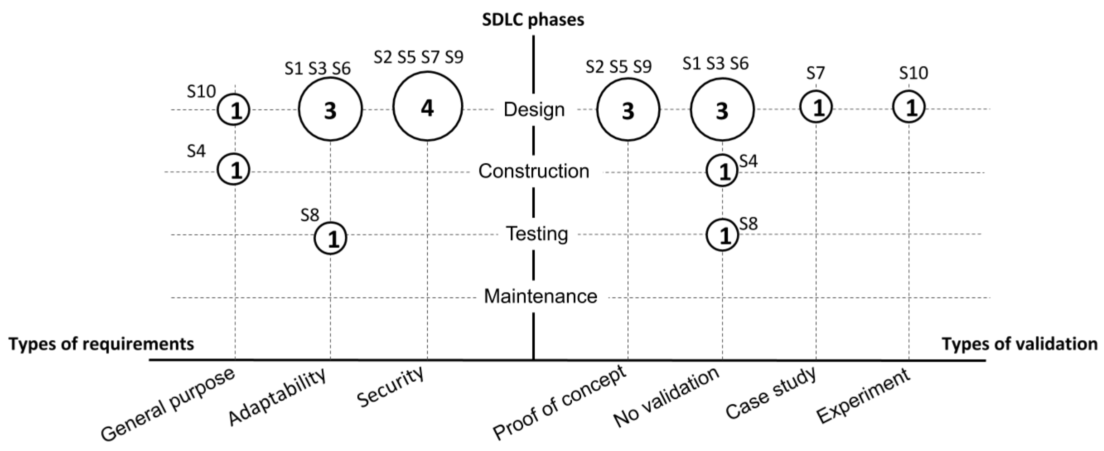

Figure 5. Mapping of the types of requirements and validation on SRPs for software design, construction, testing, and maintenance.

4.4 Discussion

Figure 5 illustrates a bubble graph that synthesizes the information we extracted and analyzed from each relevant paper.

Observe that four studies (S2, S5, S7, and S9) propose security requirement patterns with contributions to the software design phase. We conclude that this is because security is a recurrent requirement of many software systems, besides the support of well-established international standards (ISO/IEC, 2018). However, these studies mentioned above still require more significant validation with empirical assessments and use in the software industry.

Four studies (S1, S3, S6, and S8) explore SRP for the adaptability nonfunctional requirement: one in software testing (S8), and the others in software design. Besides, none of these studies presents any validation of the proposal. S4, in turn, investigates SRPs for general purpose requirements used in software construction, but also with no validation.

Still regarding Figure 5, as important as mapping the research endeavors is the analysis of the existing gaps:

1. there is a general lack of investigation on the adoption of SRPs at other SDLC stages (10), while many research endeavors still focus on requirements engineering (77);
2. adaptability and security are the most addressed nonfunctional requirements as SRPs at the software design and testing phases, from the analysis of the left-hand side of the bubble graph. However, other types of nonfunctional requirements can be specified as SRP at different SDLC phases, e.g., usability aspects with automated support for code and test case generation.
3. the application of research results on SRPs in the software industry (right-hand side of the figure); except for the studies S7 and S10, the remaining are in the proof of concept level.

5 Threats to Validity

Finding all relevant research on a topic and selecting evidence of quality are significant problems in systematic studies. Three procedures were carried out throughout the planning and the conduction phases to reduce the potential threats to the validity of this SLM.

First, we performed an automatic search strategy that combines six relevant sources of studies with search string terms based on the SEVOCAB standard vocabulary. Besides, search in the gray literature is not part of the protocol (e.g., dissertations, theses, and technical reports) because we assume that good quality research is mostly published in journals or conferences.

Secondly, we were aware that searches could be extended to two additional relevant sources of research, i.e., *ScienceDirect* and *Web of Science*. Surprisingly, the number of relevant studies resulting from the automatic search increased only from 9 to 10 (the study S10 retrieved from *Web of Science*), even introducing those two new sources. As a means of retrieving a higher number of papers, we extended the search strategy again by performing the snowballing technique over those ten relevant studies. In spite of this, this hybrid search strategy included no new research.

Thirdly, we assessed the quality of primary studies as a means of reducing a likely bias in the analysis and synthesis steps of this SLM. The quality criteria we defined and the scores we calculated for each relevant study allowed us to weight better the importance of individual studies when results were synthesised. For instance, the value of empirical evidence and the reporting of lessons learned convey a higher maturity level to the study S10 in comparison to S3. This explains somewhat the difference between their respective quality scores.

Finally, to mitigate the possibility of biases of this research, three researchers participated in the planning and conduction phases of this SLM as follows:

A: with 14 years of experience in Requirements Engineering, she performed the protocol planning, the study selection, and the data extraction and synthesis.

B: with 13 years of experience in Software Engineering, he also performed the protocol planning. Still, his contribution was mostly on the verification of the results of the selection, extraction, and synthesis activities.

C: the team leader accumulates more than 20 years of experience in Software Engineering. He helped the synthesis and writing of the results. Should divergences arose, A, B, and C solved conflicts together.

6 Final remarks

In the past few years, the literature has demonstrated the positive impacts of software requirement patterns on requirements specification quality, team productivity, elicitation and specification costs, among others (Barros-Justo et al., 2018; Irshad et al., 2018)

This paper presents a revisited version of a recent work (Kudo et al., 2019a) that investigates if those benefits from SRPs usage have also been studied for the software design, construction, testing, and maintenance. Here, we expand the scope of the search strategy with two additional and pertinent sources of studies and the application of the snowballing technique. Besides, we carry out a quality assessment activity supporting the data extraction of relevant studies.

By adding other databases in the search strategy, we obtained only one new relevant paper (S10) in comparison with our previous SLM. However, the study S10 got the highest quality score, it is the only one that defines SRP-related metrics, and it is also classified as validation research.

Concerning the overall snowballing procedure, in spite of scanning both the reference and the citation lists of the relevant studies as a means of finding further research, none of the 260 papers found suited for our purposes. This strengthens our claim that the effective use of SRPs in software design, construction, testing, and maintenance constitutes a gap for future research.

We also conclude that the studies' quality scores corroborate the maturity of each research described. The highest quality score studies (S7 and S10) achieve more empirical evidence and lessons learned than the remaining investigations about SRPs in the software design phase (studies S1 to S3, S5, S6, and S9).

In general, we are confident that our results are valuable not only for new secondary studies on this same subject but also for future primary research. To promote further research on SRPs in the whole software development process, we continue suggesting that the academic community approaches the software industry to match the latter's expectations effectively. Researchers should also establish more metrics that corroborate the advantages of SRPs usage, such as reduced design time, automatic source code generation, standardized testing, and improvement in the quality of specifications in general (Kudo et al., 2019c).

At last, we also conclude that the concrete results of the SRPs usage in practice can be better experienced through two more lines of action: SRP-based innovative development tools, and the enhancement of the current development methodologies that could integrate SRPs along the SDLC. Our current efforts include the reuse of agile concepts and practices of Behaviour-Driven Development (BDD) for the description of SRPs whose behavior is described as test patterns (Kudo et al., 2019b).

As future work, we plan the inclusion of the term "analysis pattern" (and its variants) in the search string of this systematic mapping to augment the group of relevant studies. The main reason is that analysis patterns and requirements patterns are complementary approaches (Pantoquilho et al., 2003) in such a way that the former can be transformed into the latter to migrate to the implementation details level.

Acknowledgements

This study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001. Renato Bulcão-Neto is grateful for the scholarship granted by CAPES/FAPEG (88887.305511/2018-00), in the context of the postdoctoral internship held at the Dept. of Computation and Mathematics of FFCLRP-USP. Alessandra Macedo is grateful for the financial support of FAPESP (16/13206-4) and CNPq (302031/2016-2 and 442533/2016-0). The authors would also like to thank all the anonymous referees for their valuable comments and suggestions on this paper.

References

Asnar, Y., Paja, E., and Mylopoulos, J. (2011). Modeling design patterns with description logics: A case study. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 6741 LNCS, pages 169 – 183, London, United kingdom.

Barros-Justo, J. L., Benitti, F. B. V., and Leal, A. C. (2018). Software patterns and requirements engineering activities in real-world settings:a systematic mapping study. *Comp. Standards & Interfaces*, 58:23–42.

Beckers, K., Côté, I., and Goeke, L. (2014). A catalog of security requirements patterns for the domain of cloud computing systems. In *Proceedings of the ACM Symposium on Applied Computing*, pages 337–342.

Bourque, P. and Fairley, R. E., editors (2014). *SWEBOK: Guide to the Software Engineering Body of Knowledge*. IEEE Computer Society, Los Alamitos, CA, version 3.0 edition.

Chang, F. and Gan, R. (2009). An architectural framework of the integrated transportation information service system. In *2009 IEEE International Conference on Grey Systems and Intelligent Services, GSIS 2009*, pages 1342 – 1346, Nanjing, China.

Conejero, J. M., Rodríguez-Echeverría, R., Sánchez-Figueroa, F., Linaje, M., Preciado, J. C., and Clemente, P. J. (2013). Re-engineering legacy web applications into rias by aligning modernization requirements, patterns and ria features. *Journal of Systems and Software*, 86(12):2981 – 2994.

Couto, R., Ribeiro, A. N., and Campos, J. C. (2014). Application of ontologies in identifying requirements patterns in use cases. In *Electronic Proceedings in Theoretical Computer Science, EPTCS*, volume 147, pages 62 – 76, Grenoble, France.

Curcio, K., Navarro, T., Malucelli, A., and Reinehr, S. (2018). Requirements engineering: A systematic mapping study in agile software development. *Journal of Systems and Software*, 139:32 – 50.

Fabbri, S., Silva, C., Hernandes, E. M., Octaviano, F., Thommazo, A. D., and Belgamo, A. (2016). Improvements in the start tool to better support the systematic review process. In *Proceedings of the 20th International Conference on Evaluation and Assessment in Software Engineering,* *EASE 2016, Limerick, Ireland, June 01 - 03, 2016*, pages 21:1–21:5.

Fabbri, S. C. P. F., Felizardo, K. R., Ferrari, F. C., Hernandes, E. C. M., Octaviano, F. R., Nakagawa, E. Y., and Maldonado, J. C. (2013). Externalising tacit knowledge of the systematic review process. *IET Software*, 7(6):298–307.

Franch, X., Palomares, C., Quer, C., Renault, S., and De Lazzer, F. (2010). A metamodel for software requirement patterns. In Wieringa, R. and Persson, A., editors, *Requirements Engineering: Foundation for Software Quality*, pages 85–90, Berlin, Heidelberg. Springer Berlin Heidelberg.

Hauge, A. A. and Stølen, K. (2011). SACS: A pattern language for safe adaptive control software. In *Proceedings of the 18th Conference on Pattern Languages of Programs*, PLoP '11, pages 7:1–7:22, New York, NY, USA. ACM.

Irshad, M., Petersen, K., and Poulding, S. (2018). A systematic literature review of software requirements reuse approaches. *Inf. Softw. Technol.*, 93(C):223–245.

ISO/IEC (2018). ISO/IEC 27000:2018 Information technology – Security techniques – Information security management systems – Overview and vocabulary.

ISO/IEC/IEEE (2017). ISO/IEC/IEEE 24765:2017 Systems and software engineering – Vocabulary.

Jamshidi, P., Ghafari, M., Ahmad, A., and Pahl, C. (2013). A framework for classifying and comparing architecture-centric software evolution research. In *2013 17th European Conference on Software Maintenance and Reengineering*, pages 305–314.

Kitchenham, B. A. and Brereton, P. (2013). A systematic review of systematic review process research in software engineering. *Information & Software Technology*, 55(12):2049–2075.

Kitchenham, B. A., Budgen, D., and Brereton, O. P. (2010). The value of mapping studies - A participant-observer case study. In *14th International Conference on Evaluation and Assessment in Software Engineering, EASE 2010, Keele University, UK, 12-13 April 2010*.

Knote, R., Baraki, H., Söllner, M., Geihs, K., and Leimeister, J. M. (2016). From requirement to design patterns for ubiquitous computing applications. In *Proceedings of the 21st European Conference on Pattern Languages of Programs*.

Konrad, S. and Cheng, B. H. C. (2002). Requirements patterns for embedded systems. In *Proceedings IEEE Joint International Conference on Requirements Engineering*, pages 127–136.

Kudo, T. N., Bulcão-Neto, R. F., Macedo, A. A., and Vincenzi, A. M. R. (2019a). Padrão de requisitos no ciclo de vida de software: Um mapeamento sistemático. In *Proceedings of the XXII Iberoamerican Conference on Software Engineering, CIbSE 2019, La Habana, Cuba, April 22-26, 2019.*, pages 420–433.

Kudo, T. N., Bulcão-Neto, R. F., and Vincenzi, A. M. R. (2019b). A conceptual metamodel to bridging requirement patterns to test patterns. In *Proceedings of the XXXIII Brazilian Symposium on Software Engineering, SBES 2019, Salvador, Brazil, September 23-27, 2019.*, pages 155–160.

Kudo, T. N., Bulcão Neto, R. F., and Vincenzi, A. M. R. (2019c). Requirement patterns: A tertiary study and a research agenda. *IET Software*, pages 1–9. `https://doi.org/10.1049/iet-sen.2019.0016`.

Okubo, T., Kaiya, H., and Yoshioka, N. (2011). Effective security impact analysis with patterns for software enhancement. In *2011 Sixth International Conference on Availability, Reliability and Security*, pages 527–534.

Palomares, C., Quer, C., and Franch, X. (2017). Requirements reuse and requirement patterns: a state of the practice survey. *Empirical Software Engineering*, 22(6):2719–2762.

Palomares, C., Quer, C., Franch, X., Renault, S., and Guerlain, C. (2013). A catalogue of functional software requirement patterns for the domain of content management systems. In *Proceedings of the 28th Annual ACM Symposium on Applied Computing, SAC '13, Coimbra, Portugal, March 18-22, 2013*, pages 1260–1265.

Pantoquilho, M., Raminhos, R., and Araújo, J. (2003). Analysis patterns specifications: Filling the gaps. In *Viking Plop*, Bergen, Norway.

Petersen, K., Vakkalanka, S., and Kuzniarz, L. (2015). Guidelines for conducting systematic mapping studies in software engineering: An update. *Information and Software Technology*, 64:1–18.

Tockey, S. (2015). Insanity, hiring, and the software industry. *Computer*, 48(11):96–101.

Trakhtenbrot, M. (2017). Mutation patterns for temporal requirements of reactive systems. In *Proceedings - 10th IEEE International Conference on Software Testing, Verification and Validation Workshops, ICSTW 2017*, pages 116–121.

Wen, Y., Zhao, H., and Liu, L. (2011). Analysing security requirements patterns based on problems decomposition and composition. In *2011 1st International Workshop on Requirements Patterns, RePa'11*, pages 11 – 20, Trento, Italy.

Withall, S. (2007). *Software Requirement Patterns*. Best practices. Microsoft Press, Redmond, Washington.

Wohlin, C. (2014). Guidelines for snowballing in systematic literature studies and a replication in software engineering. In *18th International Conference on Evaluation and Assessment in Software Engineering, EASE '14, London, England, United Kingdom, May 13-14, 2014*, pages 38:1–38:10.

Yang, H., Liu, K., and Li, W. (2010). Adaptive requirement-driven architecture for integrated healthcare systems. *Journal of Computers*, 5(2).