Mitigating Difficulties in Use-Case Modeling

Cristiana Pereira Bispo[1], Ana Patricia Magalhães[1,2], Sergio Fernandes[1] and Ivan Machado[3]
[1]*Post Graduated Program in Computing and Systems, Salvador University, Salvador, Brazil*
[2]*State University of Bahia, Department of Exact Sciences and Earth, Salvador, Brazil*
[3]*Federal University of Bahia, Computer Science Department, Salvador, Brazil*

Keywords: Information Systems, Strategy for Use Case Modeling, Use Case, Software Modeling, Requirement.

Abstract: The specification of software requirements in an enterprise system is crucial for software quality. The use-case (UC) approach is often used to describe software requirements because, among other benefits, its simplicity and ability to convey detailed information favors communication between business analysts, requirements analysts and, crucially, end users, who can easily understand and validate requirements. However, UC models are easier to understand than to specify, and difficulties in use case modeling (UCM) may negatively affect the quality of UC models, and so its usefulness. UC models quality can be enhanced by several modeling strategies mapped in the literature. However, no studies were found to show which of these strategies can be used to mitigate specific difficulties. There is a gap between UCM difficulties and UCM strategies. This paper presents a difficulty-strategy correlation proposal based on quality attributes of the UC model. This correlation was initially evaluated in a controlled experiment with students of an undergraduate program in computer science.

1 INTRODUCTION

Information Systems (IS) are crucial for business corporations, as they provide information that supports decision making, helping to reduce costs and improve the quality and efficiency of services or products. ISs may even enable new business models that would not be viable without them, and provide business with a competitive advantage. For all these benefits to materialize though, a precise requirements specification of an IS is of paramount importance. Properly defining the requirements of these systems is crucial for meeting the real needs of stakeholders and avoiding significant costs if requirements specification failures are identified late in the software development lifecycle.

Furthermore, even before IS deployment, during its development, requirements specification is a critical source of information for analysis and design, implementation, testing and project management. Failures in requirements specification could be easily propagated to the artifacts which use them as input, decisively influencing the success of the project.

The Use Case (UC) approach is often used to describe software requirements because, among other benefits, it favors communication between business analysts, requirements analysts and crucially end users, who can easily understand and validate requirements (Nascimento et al., 2017).

Use case modeling (UCM) has gained wide acceptance from software analysts, designers and testers (Tiwari and Gupta, 2015). The rules for creating UC models are relatively simple to use and follow. However, whether they are misapplied, it is likely that low quality UC models (Anda et al., 2001) with potentially significant impacts on the generated product would be created. The poor quality of UC models has been attributed to the inability of requirements specifiers in creating UC models. Typically, they face difficulties in both understanding and representing the requirement (Anda et al., 2009); understanding the domain of the problem (Nascimento, 2017); specifying information unambiguously (Bolloju, 2006), among others.

There are several strategies to support UCM, as discussed in (Kitchenham and Charters, 2007) as follows: i) virtualization technique for creating a conceptual mental model that represents the user's thinking of how the system works (Beimel and Kedmi-Shahar, 2018). The authors emphasize that this strategy can reduce the difficulties that affect the accuracy, completeness and redundancy of the UC

Bispo, C., Magalhães, A., Fernandes, S. and Machado, I.
Mitigating Difficulties in Use-Case Modeling.
DOI: 10.5220/0009338100430052
In *Proceedings of the 22nd International Conference on Enterprise Information Systems (ICEIS 2020) - Volume 2*, pages 43-52
ISBN: 978-989-758-423-7

model; ii) Scenario Patterns (Ko et al., 2018) that help requirements specifiers identify possible missing requirements in the UC models they have created; and iii) other strategies that use concepts of Business Process Notation (Bouzidi et al., 2017), Role-Playing (Nkamaura and Tachikawa, 2016), Antipattern (El-Attar and Miller, 2010).

Difficulties in UCM and strategies to assist UCM requirements specifiers are already mapped in the literature. However, no studies were found to show which strategies can be used to mitigate specific difficulties. Such evidence highlights a gap between difficulties and strategies, characterizing a problem that could be addressed by establishing a connection between them. The underlying hypothesis is that, if for each pointed out UCM difficulty a well-defined and tested strategy is indicated that seeks to address the lack of understanding of the requirements specifiers to model UCs, such the difficulty would be mitigated and consequently the quality of the UC models will be improved.

In this effect, the main goal of this paper is to present a proposal of correlation between difficulties in UCM and strategies for mitigating these difficulties.

Quality attributes of UC models were defined as the link between UC modeling difficulties and UC modeling strategies. In this study, we used them to create the difficulty-strategy correlation. This link emerged from the consideration that difficulties negatively affect the quality attributes of UC models, while modeling strategies positively affect the quality attributes of these models. Once a correlation between difficulties and modeling strategies was made, an assessment was performed with one of the modeling strategies to assess its effectiveness in mitigating the modeling difficulties to which it was correlated. The result shows the pertinence of the performed correlation, demonstrating that the proposed correlation can be a promising way to mitigate the difficulties that affect the requirements specifiers in the UCM.

The remainder of this paper is organized as follows: section 2 introduces the underlying concepts of this work and section 3 briefly discusses related work. Section 4 describes the proposed correlation between UCM difficulties and UCM strategies as well as provides detailed information on the methodology. Section 5 presents the correlation evaluation. Finally, section 6 draws concluding remarks and opportunities for future work.

2 BACKGROUND

Use cases are used to describe and document software requirements. They are mainly used, among other purposes, as a facilitator in the communication between project team members and others involved with the system and the use environment (Cockburn, 2000). UC modeling is the activity of designing a use-case model, which describes in detail the functional requirements of the software. They make use of graphic and textual notation to, respectively (Jacobson, 2004), i) create the UC diagram that provides a visual summary of the system services and their interaction with the environment and users (called actors); and ii) describe the interactions between the system and its actors. Difficulties regarding the syntax and semantics of graphic and textual elements in the elaboration of the UC model compromise the quality attributes (Anda et al., 2009) of this model, such as completeness, ambiguity and inconsistency.

In this paper, a *difficulty* is considered to be any lack of knowledge of requirement specifiers that prevent them from modeling UCs meeting specified quality requirements. The term *strategy* refers to any UCM resources (guidelines, procedures, or activities) proposed by researchers to improve the quality of UC models. Making a connection between difficulties and modeling strategies implies defining a relationship that either links or associates a difficulty to a strategy. However, correlating involves finding a rationale for a relationship to be conceived. The rationale investigated and identified as an effective basis for correlating UC difficulties with UC strategies are the quality attributes of the UC models defined in (Anda et al., 2009).

3 RELATED WORK

The studies deemed as related to the purpose of this research focus on the same aspect: the difficulties of UCM requirements specifiers that prevent them from building UC models meeting the defined quality requirements.

Nascimento et al. (2017) sought to explore and understand the difficulties in UCM by conducting four experimental studies. As a result, they presented a model of difficulties. Anda et al. (2006), Bolloju (2006) and Siau and Loo (2006) also investigated and reported difficulties in UCM. These works do not present any strategy to mitigate these difficulties.

To mitigate the difficulties that requirements specifiers face when modeling UCs, several authors propose the application of resources already used in other domains to verify their effectiveness in UCM. Bouzidi (2017) employed business process models to derive UCs because these models are often available in a company in the form of work instructions or administrative manuals in a clear and structured manner. Conversely, El-Atar and Miller (2012) presented an antipattern-based strategy for UCM, in which bad practices are identified to be replaced by recommended solutions. In a preceding investigation, we identified other strategies, and their respective contributions to UCM (Bispo et al., 2019). However, these studies do not indicate which strategies could actually mitigate the UCM difficulties.

The difficulty-strategy correlation proposed in this paper guides the requirements specifier in selecting the most appropriate strategy to mitigate a given difficulty. It avoids the adoption of ineffective practices, and presents various alternatives for applying tested and evaluated procedures to assist UCM.

4 CORRELATION BETWEEN UCM DIFFICULTIES AND UCM STRATEGIES

The strategies identified in the literature were proposed to improve the quality of UC models, instead of indicating which specific difficulties are mitigated by each strategy. In order to address such a concern, we defined a two-procedures methodology, as detailed next.

4.1 Correlation Methodology

The procedures adopted to make the correlation possible were two-fold: (1) obtaining a precise definition of the *meaning* of each difficulty, and then grouping them into categories; and (2) obtaining a precise definition of each quality attribute - in order to gain a deeper understanding of each attribute, in such a way that it would be possible to identify, in a UC model, which quality attributes were either met or not.

4.1.1 Categorizing UCM Difficulties

To categorize the difficulties of UCM, the studies that present these strategies, earlier presented in Bispo et al. (2019), were analyzed with the support of *Grounded Theory* (GT) (Corbin and Strauss, 2008) which helps in the construction of data-based theories.

According to Corbin and Strauss (2008), GT can be used when there is a need to understand a certain situation from a volume of information about the observed phenomenon; how and why the participants act in a certain way; and how or why a particular phenomenon or situation unfolds this way or that. An example, illustrated in Table 1, is an excerpt from one of the analyzed studies.

Table 1: A piece of text examined using GT.

"The main research question posed by this case study is whether the proposed strategy can improve the overall quality of UC models. This is achieved on two fronts: (a) by restructuring the UC diagrams to adhere to the notational syntax rules and semantics set by OMG (OMG, 2010); and (b) by changing UC descriptions to comply with recommended guidelines and widely accepted practices (Sect. 2). Therefore, the effectiveness of using our proposed approach will be assessed by comparing the resulting UC model with the original UC model, with respect to the aspects mentioned in (a) and (b)..." (El-Attar and Miller, 2010).

By using GT procedures, as Figure 1 illustrates, the highlighted phrase (taken from the example citation in Table 1) "... restructuring UC diagrams to adhere to syntax and semantic rules..." was interpreted as: difficulties that prevent UC diagrams from being modeled in accordance with syntax and semantic rules. This interpretation is supported by the information that the proposed strategy affects those aspects (syntactic and semantic rules) so that there is a general improvement in the quality of the UC models also taken from the example in Table 1.

Another set of studies were examined following the same approach, and similar interpretations were made to precisely define a set of difficulties. To this set the difficulties of UCM reported by Nascimento et al. (2017), Anda et al. (2009), Bolloju (2006) and Siau and Loo (2006) were added.

After defining the set of difficulties, they were grouped into categories, as Figure 2 shows. This is supported by our finding that a strategy which supports the identification of an UC, also supports the identification of actors and relationships. In other words, the same strategy supports the identification of the different elements (UC, actor, etc.) in the UC diagram. Therefore, it was possible to group all these difficulties in: Difficulty in identifying UCs, actors or relationships. Furthermore, considering the same example, for the word identify, some authors used the

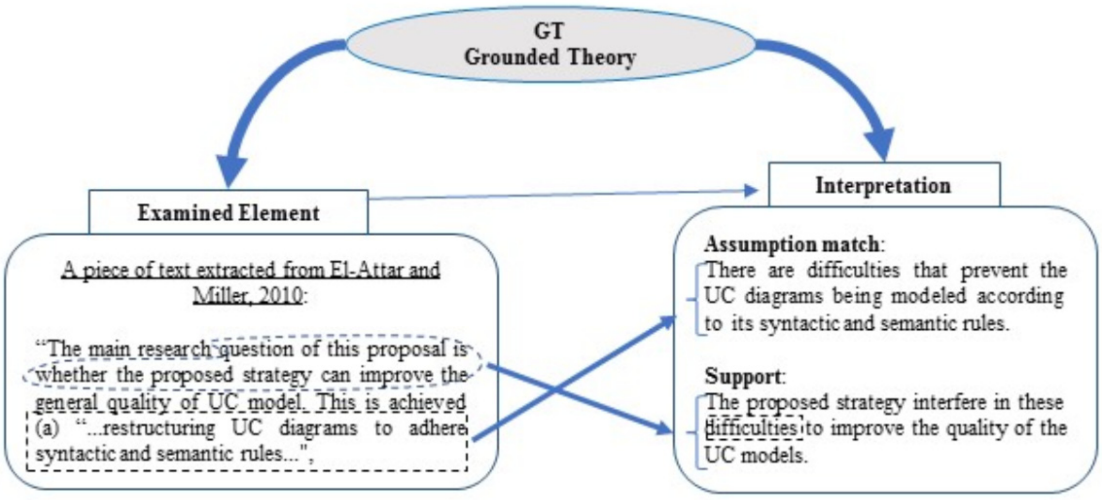

GT
Grounded Theory
Examined Element
A piece of text extracted from El-Attar and Miller, 2010:
"The main research question of this proposal is whether the proposed strategy can improve the general quality of UC model. This is achieved (a) "...restructuring UC diagrams to adhere syntactic and semantic rules...",
Interpretation
Assumption match:
There are difficulties that prevent the UC diagrams being modeled according to its syntactic and semantic rules.
Support:
The proposed strategy interfere in these difficulties to improve the quality of the UC models.

Figure 1: Using the GT method to precisely define difficulties in UCM.

word extract, while others used discover, all with the same sense of finding the UC diagram element among the requirements. The categorization considered the synonyms as well. For the difficulties illustrated in the example in Figure 2, the following category was obtained: Difficulty in identifying/extracting/ discovering UCs, actors or relationships. The same interpretation was adopted to obtain the other categories.

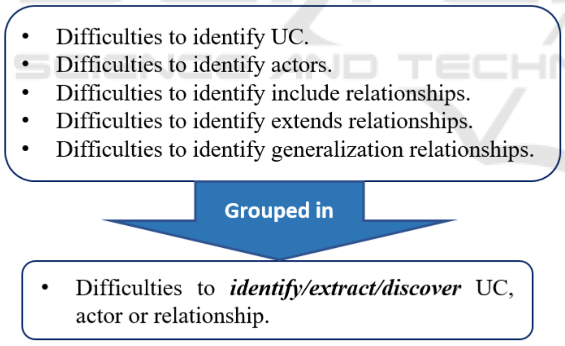

Difficulties to identify UC.
Difficulties to identify actors.
Difficulties to identify include relationships.
Difficulties to identify extends relationships.
Difficulties to identify generalization relationships.
Grouped in
Difficulties to identify/extract/discover UC, actor or relationship.

Figure 2: UCM Difficulties Grouping.

The following categories of difficulties have been established, with their respective meanings succinctly expressed:

- Difficulty in identifying/extracting/discovering UCs, actors or relationships.
  - The requirements specifier face difficulties in finding any functionality or actor; or actor and actor; actor and UC; and UC and UC relationships.
- Difficulty in representing/expressing elements of UC model.
  - This difficulty is related to the representation of relationships. For example, the requirements specifier identifies that there is a relationship (UC-actor, UC-UC, ...) but she is unsure about how to represent it. Whenever an UC extends another UC, for instance, the requirements specifier may mix up the direction of the arrow, the base case, and / or the extended case.
- Difficulty in describing/detailing the semantics of a UC model.
  - The requirements specifier is not sure about how to precisely define the meaning of any model element. For example, given the diagram, the specifier may find it difficult to define and clarify scenarios or behaviors of UCs, flows, and interactions.
- Difficulty in understanding/interpreting the problem domain.
  - The requirements specifier may find it complex to model the system considering the particularities of its actual environment. Therefore, they may leave.
  - aside important information in defining requirements.
- Difficulty in understanding implicit requirements.
  - The requirements specifier face difficulties in accurately specifying a requirement that is not explicitly defined by the domain stakeholders.
- Difficulty in synthesizing use cases.

    - The requirements specifier may not be sure about the granularity of an UC. She does not understand that she must combine correlated actions into a single function that adds value to the actor.
- Difficulty in precisely organizing the various pieces of information into the UC model.
    - The requirements specifier fails to be concise and model only what is necessary and sufficient.

UCM difficulties compromise the UC model because they affect quality attributes (Nascimento et al., 2017). Thus, it is necessary to know what a quality attribute of the UC model is and how to identify it. If it is compromised, a strategy for UCM may be indicated.

4.1.2 Refining Quality Attributes Definitions

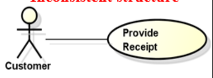
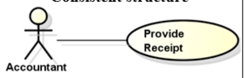
The usefulness of the UC model is a function of its quality. There are many recommendations in the literature about what *quality* means in a use case model. A list of the attributes (and their definition) from which the quality of UC models is evaluated can be found in (Tiwari and Gupta, 2015), such as completeness, consistency and ambiguity.

However, the definition of each attribute is insufficient to identify its occurrence in the UC model. This is because some definitions are not sufficiently clear and detailed, and different authors attribute different meanings to the same quality attribute. For example, the definition of the consistency attribute states that “the structure of the UCs and the use of language and grammar must be consistent across all UCs” (Anda et al., 2009). Here, the word *consistent* was used to define consistency, which does not elucidate the definition of the attribute.

A clear understanding of how the difficulties of requirements specifiers affect the quality of the UC model is required to enable the correlation proposed in this paper. To achieve this, it was necessary to refine the definition of quality attributes mentioned in the strategies for UCM. Thus, the definition of each quality attribute was extended from the definition found in the literature. This extension was based on the approach of Zayan et al. (2018) which systematically uses examples for model understanding and domain knowledge transfer. According to this approach, suitable examples help to understand subjective or abstract definitions.

Example-based Understanding Acquisition. Table 2 illustrates an example of the approach taken to clarify the definition of the consistency quality attribute.

Table 2: Example that highlights inconsistent and consistent diagrammatic structure.

Synthesis	The structure, elements, language, grammar and any information of the model (showed in the diagram in the UC description) *must have the expected semantic for them.* Must be coherent, logical and consistent.	
Word	Semantic	
Sentence	Is the perceived meaning acceptable?	
Hypothetical scenario	Soft Business	
Piece of model	**Inconsistent structure** 	**Consistent structure** 
Explanation	Syntactically there is nothing wrong: a communication relationship between an actor and a UC. However, from the point of semantics, it exists. By asking the question “Is perceived meaning acceptable?” The answer is no to the given scenario. That is, in the Soft Business scenario, Customer does not provide a receipt. Actor suggestion could be Accountant or Financial.	

The first line of Table 2 aggregates and synthesizes definitions scattered in the literature for quality attribute consistency. In the next line a keyword has been associated with the attribute to clarify its meaning. The following sentence is designed to assist the requirements specifier in judging some part of the UC model with respect to the attribute. Then we hypothesized a scenario and two simulated structures to help the requirements specifier in understanding the quality attribute.

Similarly, the same procedure was adopted for other quality attributes. Thus, a clearer and more detailed definition was constructed and is summarized as follows:

- Accuracy or Completeness or Integrity - There should be no missing information nor elements in the UC diagram and in the corresponding textual descriptions;
- Consistency - The UC model information should have the expected semantics. There should not be any conflicting elements in the diagrams and in their textual descriptions;
- Correctness - The UC diagram and its descriptions must correctly represent the requirements;
- Understandability - The information and rules contained in the UC diagrams and textual descriptions must be accurate and clearly defined;

- Ambiguity - There should be no information in the UC diagram and textual descriptions that can have more than one meaning;
- Redundancy - There should be no excessive, repetitive or superfluous information in the UC diagram and descriptions;
- Abstraction Level - The UC diagram and descriptions should present only what the software should do, at an appropriate level of granularity. That is, the UC should not be broken down into parts that have no value in themselves.

The attributes Accuracy, Completeness and Integrity of a use case model usually have the same meaning.

4.2 Link between Difficulties and Strategies for UCM

After the difficulties in UCM were categorized and quality attribute definitions for the UC model were refined, it was possible to understand how difficulties affect attributes and identify which attribute is affected (as illustrated in Figure 3).

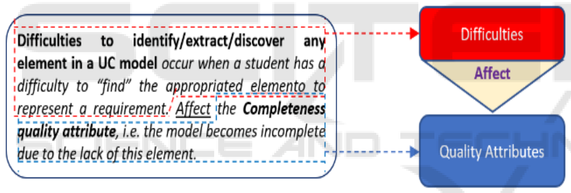

Difficulties to identify/extract/discover any element in a UC model *occur when a student has a difficulty to "find" the appropriated elemento to represent a requirement.* *Affect* *the* ***Completeness quality attribute****, i.e. the model becomes incomplete due to the lack of this element.*

Figure 3: Relationship between UCM Difficulties and Quality Attributes in UC Models.

An example of what is shown in Figure 3 is: *the difficulty to identify a UC makes the model incomplete because an expected functionality for the system will not be found*. For this example, using a strategy that supports the identification of UCs will avoid model incompleteness, nullifying the effect of the presented difficulty. Thus, the quality attributes are a link between difficulties and strategies.

Figure 4 shows the methodology for implementing the proposal to correlate an UCM difficulty with an UCM strategy.

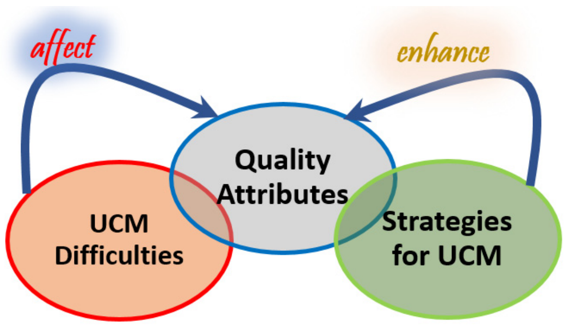


Figure 4: Quality Attributes as link between UCM Difficulties and Strategies.

4.3 Making the Correlation

The correlation, illustrated in Figure 5, is the proposed solution to the problem of lack of connection between the difficulties in UCM and the strategies for UCM that we identified.

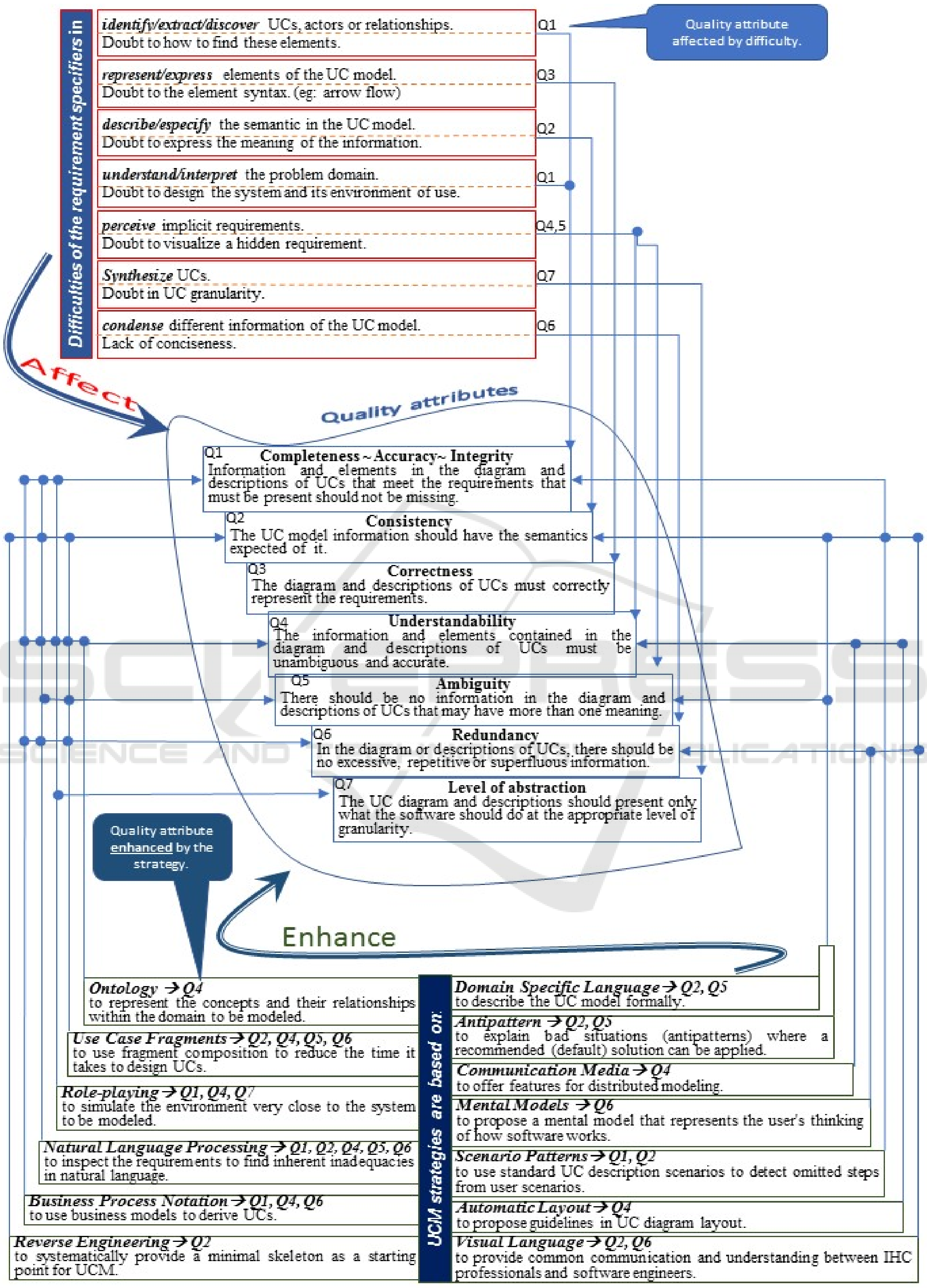
At the top of Figure 5 each difficulty is linked by an arrow to one or more quality attributes that are affected by it, in addition to the attribute code next to the difficulty. For example, next to *Difficulty identifying / extracting / discovering UCs, actors or relationships* is *Q1*, which corresponds to the quality attribute *Accuracy* or *Completeness* or *Integrity*, plus the arrow link to these attributes. (in the middle of Figure 5).

Each quality attribute, in turn, is enhanced by the use of the strategy to which it is linked to by an arrow. There is also the attribute code (s) next to the strategy name.

Example: the strategy of *Use Case Fragments* enhances the quality attributes *Q2, Q4, Q5* and *Q6*, respectively, *Consistency*, *Comprehensibility*, *Ambiguity* and *Redundancy*. Attributes *Q4* and *Q5* are affected by the same difficulty, while attributes *Q2* and *Q6* are affected by different difficulties.

Figure 5 also provides a short explanation of the meaning of each correlation element, as follows: difficulty, quality attribute, and strategy. It can be inferred from the correlation that:

- a difficulty may affect more than one quality attribute;
- when a quality attribute is affected by a difficulty, other attributes may be, as a side effect, also affected;
- a strategy can leverage more than one attribute which can be affected by the same difficulty or more than one distinct difficulty.



Difficulties of the requirement specifiers in
Quality attribute affected by difficulty.
Affect
Quality attributes
Quality attribute enhanced by the strategy.
Enhance
UCM strategies are based on:

identify/extract/discover UCs, actors or relationships.
Doubt to how to find these elements. Q1

represent/express elements of the UC model.
Doubt to the element syntax. (eg: arrow flow) Q3

describe/especify the semantic in the UC model.
Doubt to express the meaning of the information. Q2

understand/interpret the problem domain.
Doubt to design the system and its environment of use. Q1

perceive implicit requirements.
Doubt to visualize a hidden requirement. Q4,5

Synthesize UCs.
Doubt in UC granularity. Q7

condense different information of the UC model.
Lack of conciseness. Q6

Q1 **Completeness ~ Accuracy~ Integrity**
Information and elements in the diagram and descriptions of UCs that meet the requirements that must be present should not be missing.

Q2 **Consistency**
The UC model information should have the semantics expected of it.

Q3 **Correctness**
The diagram and descriptions of UCs must correctly represent the requirements.

Q4 **Understandability**
The information and elements contained in the diagram and descriptions of UCs must be unambiguous and accurate.

Q5 **Ambiguity**
There should be no information in the diagram and descriptions of UCs that may have more than one meaning.

Q6 **Redundancy**
In the diagram or descriptions of UCs, there should be no excessive, repetitive or superfluous information.

Q7 **Level of abstraction**
The UC diagram and descriptions should present only what the software should do at the appropriate level of granularity.

Ontology* → *Q4
to represent the concepts and their relationships within the domain to be modeled.

Use Case Fragments* → *Q2, Q4, Q5, Q6
to use fragment composition to reduce the time it takes to design UCs.

Role-playing* → *Q1, Q4, Q7
to simulate the environment very close to the system to be modeled.

Natural Language Processing* → *Q1, Q2, Q4, Q5, Q6
to inspect the requirements to find inherent inadequacies in natural language.

Business Process Notation* → *Q1, Q4, Q6
to use business models to derive UCs.

Reverse Engineering* → *Q2
to systematically provide a minimal skeleton as a starting point for UCM.

Domain Specific Language* → *Q2, Q5
to describe the UC model formally.

Antipattern* → *Q2, Q5
to explain bad situations (antipatterns) where a recommended (default) solution can be applied.

Communication Media* → *Q4
to offer features for distributed modeling.

Mental Models* → *Q6
to propose a mental model that represents the user's thinking of how software works.

Scenario Patterns* → *Q1, Q2
to use standard UC description scenarios to detect omitted steps from user scenarios.

Automatic Layout* → *Q4
to propose guidelines in UC diagram layout.

Visual Language* → *Q2, Q6
to provide common communication and understanding between IHC professionals and software engineers.

Figure 5: Correlation between UCM Difficulties and UCM Strategies.

5 CORRELATION EVALUATION

This section describes the correlation assessment, by reporting the goal, the research questions, the hypotheses, and the analysis and interpretation of the data.

The Antipattern-based strategy was selected to be validated because it is one of the most mentioned in the literature for UCM. As we can see in Figure 5, Antipattern enhances the consistency and ambiguity attributes. These are affected by the difficulties in describing / detailing semantics in the UC model and in understanding implicit requirements.

The purpose of the evaluation was defined according to Goal Question Metric (GQM) (Basili et al., 1994), as described next: ***To analyze*** *the Antipattern-based strategy* ***for the purpose of*** *assessing its effectiveness in mitigating the difficulties for describing / detailing semantics in the UC model and to understand implicit requirements* ***regarding*** *consistency and ambiguity,* ***from the point of view*** *of requirement specifiers*.

Based on the overall goal, the following research questions were defined (RQ):

RQ1: Is the Diagram Produced with the Support of the Strategy Free of Defects that Would Make It Inconsistent And Ambiguous? This question aimed to evaluate whether the use of the strategy corrected defects or prevented the emergence of new ones. According to Kalinowski (2012), a defect in the UC model is the failure to comply with any UC good writing rules or guidelines whose effect is to compromise some quality attribute

RQ2: Does using the Antipattern-based Strategy Mitigate the Difficulties the Requirements Specifiers Encounter that Affect the Consistency and Ambiguity of the Use Case Diagram? This question aimed to verify whether the difficulties of requirements specifiers affecting consistency and ambiguity had disappeared or reduced.

In order to conduct the evaluation a set of hypotheses were formulated: null (H0) and alternative (HA) hypotheses, illustrated in Figure 6, corresponding respectively to the existence of defects in a set of diagrams modeled without the strategy (called this set of UCD_Controlled) and another set of diagrams modeled with Antipattern-based strategy (called this set of UCD_Antipattern).

5.1 Data Analysis and Interpretation

The assessment encompassed the modeling of a similar scenario by sixteen computer science undergraduate students, which acted in the role of requirements specifiers. Each participant built two UC diagrams: one without using the strategy based on Antipatterns and another using the strategy. Inspection of the diagrams provided the results illustrated in Figure 7. As the objective of the experiment was not to evaluate the performance of the strategy, the time spent to execute the experiment was not considered. However, the effect of the strategy on the number of defects in the UCs diagram was considered.

It can be seen from Figure 7 that, with regard to ambiguity, defects were reduced from 62 to 14, and with regard to consistency, defects were reduced from 64 to 18 when using the Antipattern-based strategy. The hypotheses were assessed through the Shapiro-Wilk (1965) test, and it was possible to answer the research questions.

- Attribute ***Consistency***

  $H1_0$: The use of the antipattern strategy *does not influence* the consistency of the UC diagram.

  $Consistency\_Defect_{UCD\_Antipattern} = Consistency\_Defect_{UCD\_Controlled}$

  $H1_A$: The use of the Antipattern strategy *influences* the consistence of the UC diagram.

  $Consistency\_Defect_{UCD\_Antipattern} <> Consistency\_Defect_{UCD\_Controlled}$

  $H1_{A1}$: $Consistency\_Defect_{UCD\_Antipattern} > Consistency\_Defect_{UCD\_Controlled}$

  $H1_{A2}$: $Consistency\_Defect_{UCD\_Antipattern} < Consistency\_Defect_{UCD\_Controlled}$

- Attribute ***Ambiguity***

  $H2_0$: The use of the antipattern strategy *does not influence* the clarity of the UC diagram.

  $Ambiguity\_Defect_{UCD\_Antipattern} = Ambiguity\_Defect_{UCD\_Controlled}$

  $H2_A$: The use of the antipattern strategy *influences* the clarity of the UC diagram.

  $Ambiguity\_Defect_{UCD\_Antipattern} <> Ambiguity\_Defect_{UCD\_Controlled}$

  $H2_{A1}$: $Ambiguity\_Defect_{UCD\_Antipattern} > Ambiguity\_Defect_{UCD\_Controlled}$

  $H2_{A2}$: $Ambiguity\_Defect_{UCD\_Antipattern} < Ambiguity\_Defect_{UCD\_Controlled}$

Figure 6: Hypotheses formulated for the assessment.

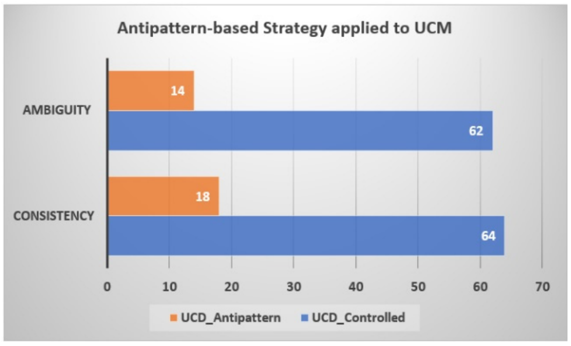


Figure 7: Effect of Antipattern-based strategy on reducing ambiguity and inconsistency of a UC diagram.

For RQ1, the use of the Antipattern-based strategy in UC diagram modeling has considerably reduced the defects that make the diagram ambiguous and inconsistent, although not all defects have been corrected.

For RQ2, in this paper we assumed that defects are generated by the UC modeling difficulties found by the requirements specifiers. Based on this assumption, if the defects that affect the consistency and ambiguity of the diagram were reduced in Antipattern modeling, then the difficulties that generated these defects have also been mitigated. However, there are limitations in the results, which are considered indicative and not conclusive.

5.2 Threats to Validity

To prevent bias in the validation we now discuss some threats to the validity of this empirical study. Regarding internal validity as the level of knowledge and experience of the participant in use case specification may influence the results of the study, we provided a training for every participant in modelling use cases. Besides, we only selected participants who were enrolled in the software engineering discipline.

Concerning external validity, to minimize the risk of sample representativeness, the diagrams produced by the participants were also inspected by people who did not participate in the experiment. The representativeness of the chosen domain as well as the size and complexity of the scenario are other threats to the experiment. As there was no possibility of using a real case, a scenario widely used in software modeling was chosen. However, experiments using real scenarios are necessary to better validate our proposal. Related to construction validity, we performed two pilot studies in order to validate the material used in the experiment. Distortions in understanding the anti-pattern strategy were minimized through a summary of examples of its use.

Finally, concerning conclusion validity, the statistic method used may influence on the conclusion. Therefore, we consulted a specialist to define which method adopt.

6 CONCLUSIONS

To mitigate the difficulties in use case modeling, this paper presented a proposal to correlate difficulties and modeling strategies, with the link between both being quality attributes of the use case model.

The correlation proposes modeling strategies to improve the quality of the UC model. Therefore, as a consequence, if a strategy improves quality, it promotes learning. If the requirements specifier learns, the difficulty is mitigated.

Thus, a preliminary assessment of a correlation triad (difficulty-attributes-strategy) was performed. The strategy tested was the one based on Antipattern and the results showed that there is clear indication that it mitigates the difficulties to which it is related: difficulty to describe / detail semantics in the UC model and to understand implicit requirements. Other difficulties may be mitigated by other strategies indicated in the correlation and which should be tested in future work.

The strategy-difficulty correlation proposed in this paper organizes and guides the requirements specifier in the selection of the most appropriate strategy to mitigate a given difficulty. This oriented indication that the correlation provides avoids the adoption of ineffective practices, as well as making the requirements specifier aware of several possibilities of applying tested and evaluated procedures to assist UCM.

REFERENCES

Anda, B., Dreiem, H., Sjøberg, D. and Jørgensen, M., 2001. Estimating software development effort based on use cases – experiences from industry. in: M. Gogolla, C. Kobryn (Eds.), UML 2001 The Unified Modeling Language. Modeling Languages, Concepts, and Tools, Lecture Notes in Computer Science, vol. 2185, Springer, Berlin Heidelberg, pp. 487–502.

Anda, B., Hansen, K. and Sand, G., 2009. An investigation of use case quality in a large safety-critical software development project. In *Information Software Technology*, vol. 51, n.12, pp. 1699–1711.

Basili, V., Caldiera, G. and Rombach, D., 1994. Goal question metric paradigm. In: Marciniak, J. (ed.)

Encyclopedia of Software Engineering, vol. 1, pp. 528–532. John Wiley & Sons, Inc., New York.

Beimel, D. and Kedmi-Shahar. E., 2018. Improving the identification of functional system requirements when novice analysts create use case diagram: the benefits of applying conceptual mental models. Requirements Engineering (2018): 1-20.

Bispo, C., Fernandes, S. and Magalhães, A. P., 2019. Strategies for Use Case Modeling: A Systematic Literature Review. In *Proceedings of the XXXIII Brazilian Symposium on Software Engineering* (SBES 2019). ACM, New York, NY, USA, 254-263. DOI: https://doi.org/10.1145/3350768.3351795.

Bolloju, N., 2006. Exploring Quality Dependencies among UML Artifacts Developed by Novice Systems Analysts. In *12th Americas Conference on Information Systems* (AMCIS 2006) pp. 472.

Bouzidi, A., Haddar, N., Abdallah, M. B. and Haddar, K., 2017. Deriving Use Case Models from BPMN Models. IEEE/ACS 14th International Conference on Computer Systems and Applications (AICCSA), Hammamet, pp. 238-243.

Cockburn, A., 2000. Writing Effective Use Cases. Reading, Addison Wesley: Massachusetts.

Corbin, J. M. and Strauss, A., 2008. Basics of Qualitative Research: Techniques and Procedures for Developing Grounded Theory. 3rd Edition. SAGE Publications.

El-Attar, M. and Miller, J., 2010. Improving the quality of use case models using antipatterns. Software & Systems Modeling, Volume 9, Issue 2, pp 141–160.

Jacobson, I., 2004. Use cases - Yesterday, today, and tomorrow. Software and System Modeling, vol. 3, No.3, pp. 210-220.

Kalinowski, M., Card, D. N. and Travassos, G.H., 2012. Evidence-Based Guidelines to Defect Causal Analysis. IEEE Software, v. 29, p. 16-18.

Kitchenham, B. and Charters, S., 2007. Guidelines for performing Systematic Literature Reviews in *Software Engineering*. version 2.3, Keele/Staffs-UK and Durham-UK.

Ko, D., Kim, S. and Park, S., 2018. Automatic recommendation to omitted steps in use case specification. Requirements Engineering.

Nascimento, E. S., Silva, W., Franca, B. B. N., Gadelha, B. and Conte, T., 2017. A Model on the Difficulties to Specify Use Cases. In *Conference Ibero-American on Software Engineering* (CIBSE), Argentina.

Nkamaura, T. and Tachikawa, Y., 2016. "Requirements engineering education using role-play training," 2016 IEEE International Conference on Teaching, Assessment, and Learning for Engineering (TALE), Bangkok, pp. 231-238.

OMG, 2010. Unified Modelling Language Superstructure - version 2.3. http://www.omg.org/spec/UML/2.3/.

Shapiro, S. S. and Wilk, M. B., 1965. An analysis of variancetest fornormality (complete samples), Biometrika 52, 591–611.

Siau, K. and Loo, P., 2006. Identifying difficulties in learning UML. Information Systems Management, vol. 23, n. 3, pp. 43-51.

Tiwari, S. and Gupta, A., 2015. A systematic literature review of use case specifications research. In *Information and Software Technology*, vol. 67, pp. 128–158.

Zayan, D., Sarkar, A., Antkiewicz, M., Maciel, R. S. P. and Czarnecki, K., 2018. Example-driven modeling: on effects of using examples on structural model comprehension, what makes them useful, and how to create them. Software & Systems Modeling, 18(3), 2213–2239. doi:10.1007/s10270-017-0652-3.