OnTheme/Doc
An Ontology-based Approach for Crosscutting Concern Identification from Software Requirements

Paulo Afonso Parreira Júnior[1, 2] and Rosângela Aparecida Dellosso Penteado[1]
[1]*Departament of Computer Science, Federal University of São Carlos, São Carlos, Brazil*
[2]*Computer Science Course, Federal University of Goiás/Campus Jataí, Jataí, Brazil*

Keywords: Aspect-Oriented Requirements Engineering, Ontologies, Early-aspects, Crosscutting Concerns.

Abstract: **Context:** Aspect-Oriented Requirements Engineering (AORE) is a research field that provides the most appropriate strategies for identification, modularization and composition of CrossCutting Concerns (CCC). **Problem:** in last years, researchers have developed several AORE approaches. However, some experimental studies have found problems with the accuracy of these approaches, regarding to the CCC identification recall. This mainly occurs, due to: (i) the lack of knowledge presented by the users of these approaches about the crosscutting nature of CCC; and (ii) the lack of resources to support users of these approaches during to CCC identification. **Goal:** this work aims to improve the values of the recall and precision metrics of a well-known AORE approach, called *Theme/Doc*, with regard to CCC identification. To do this, we propose an extension of this approach, called *OnTheme/Doc*, in which the CCC identification activity is supported by ontologies. **Experimental results:** the data obtained from an experimental study performed on *OnTheme/Doc* showed a significant increasing of recall, without negative effects on the precision and execution time of the approach.

1 INTRODUCTION

1.1 Contextualization

A set of software requirements related to the same goal/purpose is defined as a "concern" (Chitchyan et al., 2005). Ideally, each software concern should be allocated in a single module, whose responsibility is satisfying the requirements related to this concern (Dijkstra, 1976).

However, there are some kinds of concerns for which this clear allocation into modules is not possible using only the usual abstractions of software engineering, such as use cases, views-points, goals, scenarios, among others (Rashid et al., 2003). For instance, a security concern may contain requirements related to the encryption and/or access permissions control. An encryption requirement, in its turn, may affect some requirements related to orders management concern.

The previous example describes a well-known problem, called "concern tangling", that occurs when requirements of one concern affect requirements of other distinct concern(s); this problem may make hard the software understanding and evolution (Soeiro et al., 2006). Aspect-Oriented Requirements Engineering (AORE) (Araújo et al., 2004; Baniassad and Clarke, 2004; Rashid et al., 2003; Grundy, 1999) is the field that joins efforts on developing methods, techniques and tools for dealing with this problem from the initial phases of the software development cycle.

In AORE, a CrossCutting Concern - CCC (also called "Early Aspect") is a concern whose requirements affect (is tangled to) the requirements of other software concerns.

In last years, researchers have developed several AORE approaches (Parreira Júnior and Penteado, 2014). Most of these approaches include the **Concern Identification and Classification** activity, which is responsible for identifying the software concerns, as well as classifying them as base, *i.e.*, concerns that do not affect requirements of other concerns, or as crosscutting ones.

1.2 Problem and Motivation

Some experimental studies, conducted on the main

Afonso Parreira Júnior P. and Aparecida Dellosso Penteado R..
OnTheme/Doc - An Ontology-based Approach for Crosscutting Concern Identification from Software Requirements.
DOI: 10.5220/0005398301880200
In *Proceedings of the 17th International Conference on Enterprise Information Systems* (ICEIS-2015), pages 188-200
ISBN: 978-989-758-097-0

AORE approaches (Sampaio et al., 2007; Herrera et al., 2012), have pointed out the concern identification and classification as a bottleneck activity in the AORE process. The authors state that identifying CCC is harder than identifying base concerns. Some of the possible causes for this are (Sampaio et al., 2007; Herrera et al., 2012):

Base concerns are better known and understood by the scientific community than the CCC ones and many approaches are based only on the experience of software engineers who apply them. Some approaches support the software engineers during the concern identification and classification through guidelines, such as catalogues, but these guidelines generally are complex to be read and understood by humans and they are not prepared for automated semantic processing. Moreover, most of these approaches does not present a process that instruct the software engineer on how to use the proposed guidelines; and

Some CCC are not explicitly mentioned in the requirements document, *i.e.*, they emerge from other concerns and some AORE approaches are based only on searching for keywords in the requirements document, what may affect the identification of implicit concerns. For instance, if the software requires a good performance to persist its data, a possible strategy is using concurrency mechanisms, such as connection pooling. Hence, the "Concurrency" concern is observed from the existence of two other CCC: "Persistence" and "Performance" (Sampaio et al., 2007).

Theme/Doc (Clarke and Baniassad, 2005; Baniassad and Clarke, 2004) is an AORE approach that has been used, evolved and evaluated in several recent studies (Herrera et al., 2012; Ali and Kasirun, 2011; Penim and Araújo, 2010; Kit et al., 2006). This approach proposes that concern identification and classification activity be performed using a set of keywords identified by the software engineer from the software requirements. This strategy makes *Theme/Doc* highly depended on the software engineers' experience, what may lead to low levels of recall and precision, as stated in some experimental studies (Herrera et al., 2012).

1.3 Goal

The main goal of this work is increasing the recall and precision provided by the *Theme/Doc* approach, regarding to the concern identification and classification. To do this, an ontology for CCC (*OntoCCC)* and an extension of the *Theme/Doc* (*OnTheme/Doc*), in which the concern identification and classification activity is supported by the usage of *OntoCCC* instances, are proposed.

An ontology defines a specific vocabulary that captures the concepts and relationships of a domain and a set of explicit decisions (axioms), which describe the meaning of this vocabulary (Falbo et al., 2007; Guarino, 1998). Hence, the purpose of the *OntoCCC* ontology is capturing the specific concepts and relationships of crosscutting concerns domain, which have been documented in several AORE approaches in the literature (Agostinho et al., 2008; Sampaio et al., 2005; Chitchyan et al., 2006; Zheng et al., 2010; Liu et al., 2009; Soeiro et al., 2006; Moreira et al., 2005; Chernak, 2012; Whitlle and Araújo, 2004; Alencar et al., 2010; Brito and Moreira, 2003; Mussbacher et al., 2010).

In this work, we consider that the usage of *OntoCCC* ontology may improve the recall and precision provided by the *Theme/Doc* approach as follows:

Regarding to the Dependence of Software Engineers' Experience: the knowledge base of the *OntoCCC* ontology may be used for the definition of better keywords, aiming to minimize the dependence of the professionals' experience; and

Regarding to the Implicit CCC: the mutual influence that exists between different CCC may be documented in the *OntoCCC* ontology, aiming to allow the identification of CCC that are not explicitly described in the requirements document.

It is important to state that the usage of ontologies in the context of requirements engineering has been widely explored (López et al., 2008). However, according to a recent systematic mapping of the literature, conducted by the authors of this paper, the usage of ontologies for concern identification and classification has not been fully exploited yet.

1.4 Evaluation and Contributions

To verify the goal proposed in this paper, an experimental study involving undergraduate and graduate students in Computer Science from two Federal Universities in Brazil was conducted.

The study was planning and implemented according to the procedure proposed by Wohlin et al., (2012). As results, it was observed that, with 99.9% of significance level, the values of recall provided by the *OnTheme/Doc* approach is higher than those provided by *Theme/Doc*, regarding to the CCC identification. Besides, it was observed no significant differences with regard to the precision provided by both approaches, neither for the time

spent by the participants of the experiment during the application of these approaches.

The main contributions of the paper are threefold: (i) the proposed ontology can help software engineers to better understand the main concepts and relationships of CCC; (ii) the proposed ontology can allow software engineers to perform automatic processing on it, what it is not easy to do with other kinds of approaches, like catalogues, vocabularies, thesaurus, among others; and (iii) the proposed approach presents how to use ontologies in the context of CCC identification and gives indications that this usage can improve the accuracy of this activity.

This paper is organized as follows: (i) Section 2 presents the main concepts about domain ontologies and the *Theme/Doc* approach; (ii) Section 3 describes the *OnTheme/Doc* approach, as well as the *OntoCCC* ontology; (iii) in Section 4, the planning, execution, results and threats to validity of the experimental study performed in this work are presented; (iv) Section 5 discusses the main works related to the proposal of this paper; and (v) finally, Section 6 presents the final remarks of this work and some proposals for future works.

2 BACKGROUND

2.1 Ontologies

A domain ontology can be defined as a simplified and abstract view of a domain that includes the concepts of some area of interest and the relationships between them (Gruber, 1995; Fensel, 2001). One important feature of an ontology is its must be shared, *i.e.*, the knowledge captured by an ontology must be consensual, not limited to a specific individual. This section presents the three main concepts of domain ontologies: **Classes**, **Properties** and **Individuals** (Horrige et al., 2011; Hernandes, 2009; Lima and Carvalho, 2005). To illustrate these concepts, parts of the *OntoCCC* ontology (Section 3) are presented.

Classes are concrete representations of a concept. In practical terms, classes are interpreted as sets that contain individuals (Horrige et al., 2011). For example, the *CCC* class (represent by an oval shape in Figure 1) represents all individuals that are crosscutting concerns. Classes can be arranged in superclass-subclass hierarchies. In Figure 1, the *FunctionalCCC* and *NonFunctionalCCC* classes are subclasses of *CCC* (described by the "is-a" relationship), what means that all functional and non-functional CCC also are crosscutting concerns.

Properties are binary relationships that connect two individuals, two classes, an individual and a value or a class and a value. There are two main kinds of properties: "Object Properties" and "DataType Properties". The "Object Properties" are used to define relationships between classes. For example, the *hasKeywords* property (Figure 1) connects the *CCC* class to the *Keywords* class. This property defines that a crosscutting concern, functional or non-functional, may contain a set of keywords that can be used to identify it. The *hasSubconcerns* property indicates that a CCC can be decomposed into sub-concerns, which also are CCC. Similarly, the *hasSynonymes* property recursively connects the *Keywords* class to itself, representing that a keyword may contain synonyms.

A "DataType Property" connects a class to a primitive value (*e.g.* an integer or a string value). For example, the *CCC* class has the *name* and *description* properties (Figure 1), which can be connected to strings values; these properties specify, respectively, the name of a particular CCC and its description.

It is possible to enhance the meaning of the properties through the usage of attributes, such as "transitivity", "symmetry", among others. Due to space limitations, the definition of them was omitted; more details can be found in Horrige *et al.* (2011).

Individuals, also known as "instances" or "class instances", represent objects of the domain of interest. In the *OntoCCC* ontology, examples of individuals are instances of CCC already identified and well-known by the scientific community (for example, security, logging, among others).

An example with six individuals is illustrated in Figure 2: four individuals are instances of the *NonFunctionalCCC* class and two of the *Keywords* class (classes are highlighted in gray and individuals, in white). In this example, the non-functional CCC are "Logging", "Persistence", "Connection" and "Transaction"; "Connection" and "Transaction" are sub-concerns of "Persistence". In addition, the "Logging" concern is related to two keywords, called "logged" and "log", which are synonymous.

2.2 *Theme/Doc* Approach

The *Theme* approach supports the AORE field in two levels. At requirements level, the approach is called *Theme/Doc* and allows the software engineers

to: (i) identify the software concerns from a set of keywords and software requirements; and (ii) refine the views provided by the approach to reveal which concerns are base and which are crosscutting ones. At design level, the approach is called *Theme/UML* and allows the software engineers to model, through specific notations, base and crosscutting concerns and specify how they can be combined.

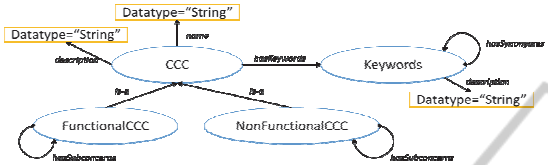

Figure 1: Part of the *OntoCCC* ontology.

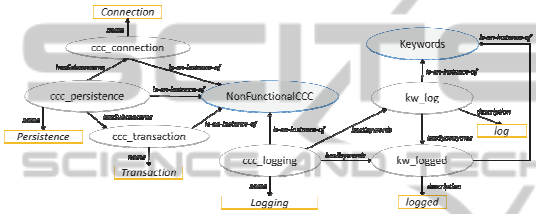

Figure 2: Instantiation of the *OntoCCC* ontology.

The focus of this paper is on concern identification and classification activity. Hence, to illustrate the main features of the *Theme/Doc* approach, an example of a Course Management Software – CMS (Baniassad and Clarke, 2004), whose requirements are outlined in Table 1, is used.

To identify concerns, *Theme/Doc* offers a visualization resource, called "action-view". Two inputs are required to build an action-view: (i) a list of key-actions, which consists of verbs identified by the software engineer from the software requirements; and (ii) a set of software requirements. Based on these inputs, the software engineer performs an analysis of the requirements document and generates a preliminary action-view.

Table 1: Requirements description of a course management software (Baniassad and Clarke, 2004).

#	Requirements Description
R1	Students can **register** for courses.
R2	Students can **unregister** for courses.
R3	When a student registers then it must be **logged** in their record.
R4	When a student unregisters it must also be logged.
R5	Professors can unregister students.
R6	When a professor unregisters a student it must be logged.
R7	Professors can **give** marks for courses.
R8	When a professor gives a mark this must be logged in the record.

A preliminary action-view is a view in which actions were not classified as base or crosscutting ones yet. Figure 3 illustrates the preliminary action-view created from the requirements and the list of key-actions of the CMS software (highlighted words, in the text of Table 1). The key-actions are represented by diamonds and the requirements by boxes with rounded edges.

If a requirement contains a key-action in its description, then it is associated with this action by an arrow that starts in the requirement and ends in the key-action. The set of key-actions should be identified by the software engineer based on his/her experience with regard to the domain for which the software is being developed. There are situations where a requirement may refer to more than one key-action. For instance, the requirement "R3" (Figure 3) refers to *register* and *logged* actions. In *Theme/Doc* approach, CCC are identified by analyzing such requirements.

To perform the classification of actions as base or crosscutting ones, the preliminary action-view and the set of software requirements are required. The software engineer initially must examine the requirements that refer to more than one action and determine what is the primary action (more important action) of these requirements. In the case of requirement "R3", the primary action is *logged*, since the requirement was written to specify the implementation of logging behavior. As the *register* action is not the primary action of this requirement, we say that this action is being affected by the behavior of the *logged* action. Hence, the *logged* action is classified as a crosscutting action and *register*, as a base action. To represent this kind of information, an arrow with a point at one of its ends is drawn from the *logged* action to the *register*, indicating that *logged* affects the *register* action.

The software engineer should examine all requirements that share the *logged* action and decide if they also are affected by its behavior. In addition, the software engineer should keep on examining the other requirements that share more than one action. Finally, after analyzing all requirements and actions, an extended action-view is generated (Figure 4), with three base actions (*unregister*, *give* and *register*) and one crosscutting action (*logged*) that cut-across all the three base actions. The main strengths of the *Theme/*Doc approach are: (i) it uses visualization resources as a strategy for concern identification, what allows the software engineering to have a better view of the software concerns; (ii) it is independent of the requirements document language; and (iii) it has been widely used, evolved

and evaluated in recent works (Herrera et al., 2012; Ali and Kasirun, 2011; Penim and Araújo, 2010; Kit et al., 2006).

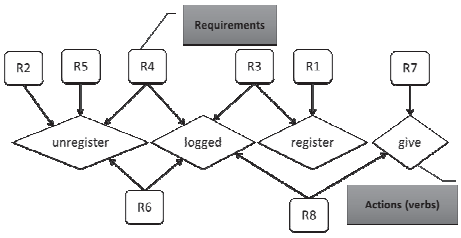

Figure 3: Preliminary action-view.

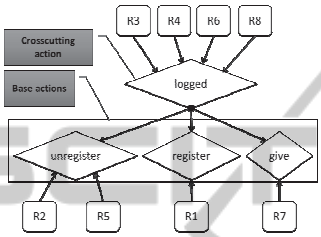

Figure 4: Extended action-view.

As limitations, it is possible to note that *Theme/Doc*: **depends on the usage of keywords**; **depends on the software engineers' experience;** and **does not support the software engineering during the identification of implicit concerns**.

Section 3 of this paper presents an extension of the *Theme/Doc* approach, called *OnTheme/Doc*, as well as an ontology for CCC, called *OntoCCC*. The main goal of the *OnTheme/Doc* and *OntoCCC* is to minimize the weaknesses of *Theme/Doc* and, hence, to improve the values of recall and precision provide by this approach.

3 *OnTheme/Doc* APPROACH

As described in Section 2, *Theme/Doc* requires that the software engineer works using only his/her prior knowledge about the problem domain and the concepts of concern identification and classification. This makes the approach highly dependent on the experience of its users.

According to the extension proposed in this paper, besides his/her prior experience, the software engineer has the support of the knowledge represented in one or more instances of the *OntoCCC* ontology.

It is important to note that although *Theme/Doc* supports the identification of base and crosscutting concerns, this paper is worried only with the CCC identification, because, as already stated in this paper, this has been the bottleneck in the AORE process (Sampaio *et al.*, 2007; Herrera *et al*., 2012).

3.1 *OntoCCC* Ontology

OntoCCC ontology is responsible for representing well-known and already published concepts and relationships on CCC. The concepts and relationships of the *OntoCCC* ontology describes the main features of a CCC, such as the name commonly used to identify it in the scientific community, its description, if it is a functional or non-functional CCC, as well as its possible relationships with other concerns.

To build the *OntoCCC* ontology, several studies that addressed the concern identification and classification subject were analyzed (Agostinho et al., 2008; Sampaio et al., 2005; Chitchyan et al., 2006; Zheng et al., 2010; Liu et al., 2009; Soeiro et al., 2006; Moreira et al., 2005; Chernak, 2012; Whitlle and Araújo, 2004; Alencar et al., 2010; Brito and Moreira, 2003; Mussbacher et al., 2010).

In a systematic mapping conducted by the authors of this paper (Parreira Júnior and Penteado, 2014), thirty-eight AORE approaches were identified. Among these, ten included the concern identification and classification activity and proposed the usage of guidelines to support the software engineers during the execution of this activity. The main features of these guidelines were used to construct the *OntoCCC* ontology; this was performed to guarantee that the *OntoCCC* captures a consensual knowledge.

For each concept/relationship defined in *OntoCCC*, we describe what work served as inspiration for it. The full version of *OntoCCC* ontology is presented in Figure 5.

The concepts represented by *CCC*, *FunctionalCCC*, *NonFunctionalCCC* and *Keywords* classes, as well as the *name*, *description*, *hasSubconcerns*, *hasKeywords* and *hasSynomymes* properties were briefly commented in Section 2.1.

The *CCC* and *NonFunctionalCCC* concepts are well-known in AORE community and are reported in all analyzed studies. The *FunctionalCCC* concept, however, was taken from work of Moreira *et al.* (2005), which was the first study to report that functional requirements also can cut-across other software requirements. Hence, the *FunctionalCCC* class represents the concerns related to functional features of the software that cut-across requirements of other concerns, for example, "Orders Management" and "Virtual Shopping Cart".

The concept of keywords appears in several AORE approaches (Agostinho et al., 2008; Sampaio et al., 2005; Chitchyan et al., 2006), including the *Theme/Doc*.

However, only the proposal of Agostinho et al., (2008) presented a template to store the keywords used for identifying specific concerns.

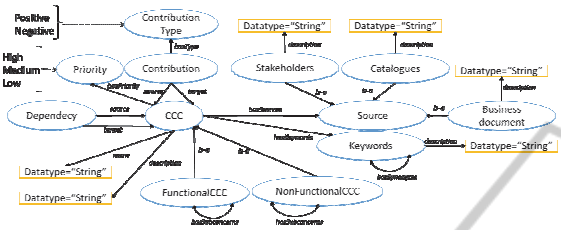

Figure 5: *OntoCCC* ontology.

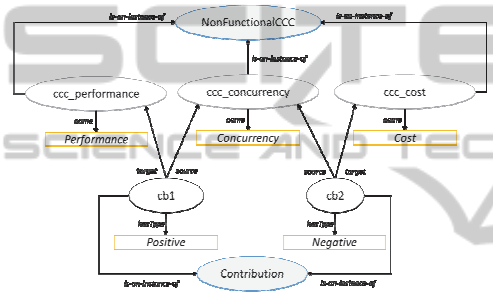

Figure 6: Contributions between different CCC.

It is important because these keywords contain knowledge about these concerns that can be reused for concern identification in future projects.

The *Keywords* class was designed to store the keywords (and its synonyms) commonly used to identify a particular CCC. The *hasSynonyms* property can be useful when the software engineer wants to know how many distinct words (not synonymous) are present in the requirements document. We believe the more distinct keywords about a CCC presented in a requirements document, the stronger the indications of the presence of this CCC in the software.

The idea of decomposing concerns into sub-concerns, represented by the *hasSubconcerns* property, is new one and was not found in the analyzed studies. This property was considered important, since a given concern may be too large and complex that may complicate the reasoning of the software engineer. Hence, by decreasing the granularity of these concerns, treating them as sub-concerns, it is possible to know what kinds of concerns really are in the software and what are the most appropriate strategies to modularize them.

The concept represented by the *Source* class appears in some AORE approaches, such as proposed by Agostinho et al., (2008), Moreira et al., (2005) and Whitlle and Araújo (2004). A source can be: (i) a suggestion of a stakeholder, *e.g.* the project manager – *Stakeholder* class; (ii) a catalog, for instance, the catalog of non-functional requirements proposed by Chung and Leite (2000) – *Catalogues* class; or (iii) a business document, such as a security protocol of a company, among others – *Business Document* class. A CCC may be related to several sources through the *hasSources* property. Each kind of source has a *description* property that may store more information on it.

The two main types of relationships between CCC are defined by *Dependency* and *Contribution* classes. *Dependency* class defines a dependency relationship between two CCC: a source and a target. This means if "A" (source) depends on "B" (target) and "A" appears in the software requirements document, then "B" need to be there too. This type of information is important because: (i) it allows the software engineer to explore other CCC, before unrecognized by him/her, *i.e.*, by saying that "A" depends on "B", he/she should also look for keywords related to "B" concern in the requirements document; and (ii) it allows the software engineer to verify inconsistencies in the requirements document, because, if a CCC "A" depends on "B" and "B" is not described in the software requirements and is not an implicit concern, then the requirements document may be inconsistent.

Another important concept about CCC is represented by the *Contribution* class. It represents a mutual influence between different CCC. This kind of influence is reported in the catalog Chung and Leite (2000), but only for non-functional requirements. In AORE field, Moreira et al., (2005) address this type of influence on their work. To do this, the authors proposed a "contribution matrix", which is created by the software engineer, based on his/her experience and on some catalogues of non-functional requirements. In this matrix, it is possible to visualize the kind of contributions (negative or positive) between different CCC of the software. However, the knowledge about the contribution between several CCC is limited to the project under analysis and there are no clearly defined mechanisms to reuse it in later projects.

A contribution can be *Negative* or *Positive* as defined by the *ContributionType* class and the *hasType* property. For example, the "Information Retrieval" and "Mobility" concerns are related as

follows (Moreira et al., 2005): the higher the mobility, the greater the difficulties of retrieving information. This means that "Mobility" negatively contributes to "Information Retrieval". The inverse contribution is also negative, since the more complex is the information to be retrieved, the less mobile the software can be, since some wireless networks have limited bandwidth size.

Another example is the case of the "Concurrency", "Performance" and "Cost" concerns. The implementation of concurrency mechanisms in the software can positively contribute to the software performance, but not to the cost of the project.

The knowledge presented in both previous examples can be represented in the *OntoCCC* ontology by means of *Contribution*, *ContributionType* and *CCC* classes and *hasType*, *source* and *target* properties. Figure 6 presents an instance of the *OntoCCC* ontology, in which the contribution between "Concurrency", "Performance" and "Cost" concerns is presented.

Finally, each CCC has a *hasPriority* property that relates a CCC to an instance of the *Priority* class. The priority can be defined by stakeholders or experts in CCC and may assume the following values: "High", "Medium" or "Low". This information is important when one specific CCC "A" is negatively influenced by other two different CCC "B" and "C", or when one specific CCC "A" exerts negative and positive influences on two different CCC "B" and "C"; in these cases, the software engineer must decide on what concern will be addressed and he/she need to know what are the impacts of his/her decision.

In the example of "Concurrency", "Performance" and "Cost" concerns, the software engineer will have to decide between prioritizing cost or performance; the *Priority* class and the *hasPriority* property may provide more information for the software engineer to make his/her decision. Priority is a concept discussed in the work of Moreira et al., (2005), but it is used only in the conflict detection and resolution activity – one of the last activities in the AORE process. We believe that treating this issue in the beginning of the AORE process is important, because it can reduce the rework, as well as the propagation of errors throughout this process.

Using the concepts and relationships of *OntoCCC* ontology, commented above, it is possible to store the existing knowledge about specific types of CCC, creating instances of this ontology. Small examples of *OntoCCC* instances for the "Persistence", "Connection", "Transaction", "Logging", "Concurrency", "Performance" and "Cost" concerns were described in Figure 2 and Figure 6.

Instances of *OntoCCC* ontology can be created from: (i) catalogues of crosscutting concerns; (ii) other kind of catalogues, *e.g.*, the catalogue of non-functional requirements, such as those proposed by Cysneiros (2014) and Chung and Leite (2000); (iii) the knowledge of experts on AORE; or (iv) historical data of previous projects, among others.

3.2 *OnTheme/Doc*

We believe that the knowledge represented by the instances of the *OntoCCC* ontology may help the software engineers to perform the concern identification and classification activity in a more effective way. Hence, it was proposed an extension of *Theme/Doc* approach, called *OnTheme/Doc*.

The execution of *OnTheme/Doc* approach follows the same flow of the *Theme/Doc*. However, there are two new activities to be performed by the software engineers ("Checking Dependencies" and "Checking Contributions"), and the procedure for key-actions identification was redefined.

"Identifying Key-actions": the software engineer should analyze each CCC defined in the *OntoCCC* instance, searching for the keywords presented in this instance in the requirements document.

"Checking Dependencies": for each identified CCC, the software engineer should verify the relationships of it with other CCC, in order to detect possible dependencies between them. If there are dependencies between a CCC "A" with "B" e "C", the software engineer should also consider the keywords of "B" and "C". If there are no keywords related to "B" and "C" in the requirements, they may be implicit concerns and should be analyzed in the "Checking Contributions" activity, or the requirements document is inconsistent; and

"Checking Contributions": after building an action-view, the software engineer should analyze the CCC ontology again looking for possible contributions of a CCC over other ones. In this activity, new CCC, before unidentified, may appear due to the mutual influence between different CCC. In addition, it may be necessary to resolve conflicts between different CCC. For this, the value of the priority property of each conflicting CCC must be observed. If the conflict persists (for example, when the priority levels of two CCC are the same), meetings with stakeholders may be necessary.

4 EXPERIMENTAL STUDY

The evaluation goal of this work is: "**To analyze:** the *OnTheme/Doc* approach. **In order to:** evaluate. **With respect to:** recall and precision provided by this approach. **From the point of view of:** software engineers. **In the context of:** a group of undergraduates and graduate in Computer Science.".

4.1 Planning

The planning of this experimental study was defined according to the Wohlin's proposal (Wohlin et al., 2012) and involves the following steps: (i) context selection; (ii) hypotheses formulation; (iii) variables selection; (iv) participants selection; and (v) design and execution of the experimental study.

a) Context Selection. This experimental study was conducted with fourteen undergraduate and graduate students in Computer Science from two Federal Universities in Brazil.

An information system that aims to record complaints in health area, called *Health Watcher* (2014), was used in this study. It is a well-known application in the AORE field and was chosen because it has a suitable requirements document for CCC identification. Its requirements document presents several CCC, such as security, persistence, concurrency, among others. In addition, all CCC of this application have already been identified and cataloged by experts (Health Watcher, 2014), serving as an oracle to verify the answers given by the participants of this experimental study.

b) Hypotheses Formulation. An important part of an experimental study is to specify the metrics that will be used. Based on these metrics, the researcher may establish hypotheses and draw conclusions from the results of the experiment.

In this work, three metrics were used, whose formulas and description are presented in Table 2: recall, precision and f-Measure (a harmonized average of the recall and precision).

These metrics are commonly used for measuring the effectiveness of products and processes in several research areas, such as information retrieval, natural language processing, among others. They also are widely used at works on concern identification and classification (Herrera *et al.*, 2012; Sampaio *et al*., 2007). In this work, the interpretation of these metrics is very trivial, the higher the value of recall, precision and f-Measure, the better the effectiveness of the approach. Based on these metrics, six hypotheses were developed for this study, two related to the recall metric, two for precision and two for f-Measure (Table 3).

Table 2: Metrics of the experimental study.

Metrics		
Recall (Re)	**Precision (Pr)**	**f-Measure (fM)**
$\text{Re} = \left(\frac{CIC}{EC}\right) * 100$	$\text{Pr} = \left(\frac{CIC}{TIC}\right) * 100$	$fM = 2 * \left(\frac{\text{Re} * \text{Pr}}{\text{Re} + \text{Pr}}\right)$
Description		
CIC (*Correctly Identified Concerns*): amount of correctly identified concerns, *i.e*., without the false positives. **Re (Recall)**: percentage of correctly identified concerns, regarding to the amount of existing concerns. **EC (*Existing Concerns*)**: amount of existing concerns. **TIC (*Total of Identified Concerns*)**: amount of identified concern, *i.e*., including the false positives. **Pr (Precision)**: percentage of correctly identified concerns, regarding to the amount of identified concerns. **fM (f-Measure)**: a harmonized average of the recall and precision values.		

c) Variables and Participants Selection. Independent variables are those manipulated and controlled during the experimental study. In this study, the independent variable is related to the approaches for concern identification and classification. The dependent variables are those under evaluation and whose variations must be observed. In this experiment the recall, precision and f-Measure metrics are considered as dependent variables. The participants of this study were selected through non-probability for convenience sampling.

d) Design and Execution of the Experimental Study. The distribution of the participants was performed aiming to form two homogeneous groups, with regard to the participants' experience and the amount of available participants in each group. Each group had seven participants and the participants' experience was verified by the application of a profile characterization questionnaire. It takes into account the knowledge of the participants about AORE and *Theme/Doc* approach. In addition, the experimental study was planned in phases (training and execution) to minimize the effect of participants' knowledge of the dependent variables.

Before starting the execution of the experimental study, a training was conducted, in order to homogenize the knowledge of participants on AORE and *Theme/Doc* and *OnTheme/Doc* approaches. During the training, it was not informed to the participants what approach was developed by the authors of this paper.

Table 3: Hypotheses of the experimental study.

Hypotheses for Recall	Hypotheses for Precision	Hypotheses for f-Measure
H_{0Re}: there is no difference of using *OnTheme/Doc* or *Theme/Doc*, regarding to the recall. H_{0Re}: $Re_{OnThD} = Re_{ThD}$	H_{0Pr}: there is no difference of using *OnTheme/Doc* or *Theme/Doc*, regarding to the precision. H_{0Pr}: $Pr_{OnThD} = Pr_{ThD}$	H_{0fM}: there is no difference of using *OnTheme/Doc* or *Theme/Doc*, regarding to the f-Measure metric. H_{0fM}: $fM_{OnThD} = fM_{ThD}$
H_{1Re}: there is difference of using *OnTheme/Doc* or *Theme/Doc*, regarding to the recall. H_{1Re}: $Re_{OnThD} \neq Re_{ThD}$.	H_{1Pr}: there is difference of using *OnTheme/Doc* or *Theme/Doc*, regarding to the precision. H_{1Pr}: $Pr_{OnThD} \neq Pr_{ThD}$.	H_{1fM}: there is difference of using *OnTheme/Doc* or *Theme/Doc*, regarding to the f-Measure metric. H_{1fM}: $fM_{OnThD} \neq fM_{ThD}$.
X_{OnThD}, where X is a metric, means: the value of X obtained by a specific participant using the *OnTheme/Doc* approach. **X_{ThD}**, where X is a metric, means: the value of X obtained by a specific participant using the *Theme/Doc* approach.		

In the execution phase, the participants had to identify the CCC existing in the requirements document of the *Health Watcher* application. To do this, the Group 1 used the *Theme/Doc* approach and the Group 2, the *OnTheme/Doc*. The part of the requirements document analyzed by the participants had seven types of non-functional CCC: "Security", "Concurrency", "Usability", "Performance", "Distribution", "Availability" and "Persistence". "Distribution" and "Competition" were implicit concerns, *i.e.*, there were not keywords in the requirements document with regard to them.To calculate the values of the recall, precision and f-Measure metrics, it was considered the amount of CCC identified by each participant, individually.

The participants of the Group 2 also received an instance of the *OntoCCC* ontology, created by the authors of this paper, from the catalogs of Moreira et al., (2005), Chung and Leite (2000) and Cysneiros (2014). This instance was omitted of this paper due to the limitation of space, but can be found at https://db.tt/Wqx2xWh3.

4.2 Results

Table 4 presents the results of this experimental study. The first (lines 1-10) and the second (lines 11-20) parts of this table, respectively, present the results for the *Theme/Doc* and *OnTheme/Doc* approaches. The first column presents the codes that identify each participant; the second, third and fourth columns refer to the values of recall, precision and f-Measure; the last column of this table shows the time (in minutes) that each participant took to finalize the CCC identification.

The values for the recall, precision and f-Measure metrics emphasize a statement made by Sampaio *et al*. (2007) in their experimental study on AORE approaches: "Generally the AORE approaches do have good precision (…). However, the majority of these approaches do have limitations when considering recall". This means that there is little incidence of false positives, but the amount of correctly identified concerns is low. The precision values of both groups were higher than the recall values.

Table 4: Experimental results.

Approach *Theme/Doc*				
Participant	**Re (%)**	**Pr (%)**	**fM (%)**	**Time (min)**
P1	42,85	80,00	55,80	43
P2	42,85	100,00	59,99	48
P3	42,85	100,00	59,99	49
P4	28,57	80,00	41,48	48
P5	57,14	80,00	66,66	36
P6	42,85	100,00	59,99	31
P7	28,57	100,00	44,44	34
Average	**40,81**	**91,42**	**55,48**	**41**
Approach: *OnTheme/Doc*				
Participant	**Re**	**Pr**	**fM**	**Time (min)**
P8	71,42	80,00	75,47	62
P9	85,71	100,00	92,30	39
P10	85,71	100,00	92,30	54
P11	71,42	100,00	83,32	37
P12	57,14	80,00	66,66	43
P13	71,42	80,00	75,47	42
P14	71,42	100,00	83,32	42
Average	**73,46**	**91,42**	**81,26**	**45**

Taking into account the values for recall, participants who used the *OnTheme/Doc* approach had, on average, more promising results than those who used the *Theme/Doc*.

Table 5: Concerns identified by each participant.

#	Participants *Theme/Doc*							%	Participants *OnTheme/Doc*							%
	1	**2**	**3**	**4**	**5**	**6**	**7**		**8**	**9**	**10**	**11**	**12**	**13**	**14**	
1	X	X						28	X	X	X	X	X			57
2	X	X	X	X	X	X	X	100	X	X	X	X	X	X	X	100
3		X						14		X	X	X		X	X	71
4			X	X	X	X		57	X	X	X	X		X	X	85
5					X	X	X	43	X	X	X			X	X	71
6	X		X		X			43	X	X	X	X	X			71
7								0					X	X	X	43
Average								**41**	**Average**							**71**
Legend: (1) Persistence; (2) Security; (3) Concurrency; (4) Usability; (5) Performance; (6) Availability; (7) Distribution																

To improve the discussion about the recall

values, Table 5 presents: (i) the list of CCC of the *Health Watcher* application - first column; (ii) the CCC identified by each participant who used the *Theme/Doc* approach - from second to eighth columns; (iii) the percentage of participants who identified each CCC - ninth column; and (iv) the same information previously described to the *OnTheme/Doc* approach – from tenth to the eighteenth columns.

Based on this table, it is possible to note that only one of the participants who used the *Theme/Doc* approach was able to identify the "Concurrency" concern and none of them has identified the "Distribution" concern; "Concurrency" and "Distribution" were implicit concerns. Regarding to the participants who used *OnTheme/Doc* approach, just one participant did not identify the two implicit concerns. For all concerns, the percentage of participants who identified them is always greater for *OnTheme/Doc* approach than for the *Theme/Doc*. Consequently, on average, the percentage of participants who identified any concern using *OnTheme/Doc* approach (71%) is higher than that one who used *Theme/Doc* (41%).

Finally, it is important to note that even using ontologies, the percentage of participants who identified the "Distribution" concern is not satisfactory (43%). This indicates that the strategy used to represent the mutual influence between different concerns must be reviewed.

Based on Table 4 again, it is possible to note that there is no difference between the two approaches with regard to the precision. This means that there was not a high incidence of false positives during the CCC identification for both approaches.

Regarding to f-Measure metric (Table 4), the average value obtained by the participants who used *OnTheme/Doc* was higher than that one obtained to the *Theme/Doc* approach. This occurs, because the precision provided by the two approaches is similar and the recall provided by *OnTheme/Doc* approach is higher than that one provided by *Theme/Doc*.

Table 4 still presents that the average time for execution of *OnTheme/Doc* (45 min) was higher than that one provided by *Theme/Doc* approach (41 min). This is due to the participants who used the *OnTheme/Doc* approach had another artefact to analyzed, *i.e.*, the instance of the *OntoCCC* ontology, as well as two new activities to be performed: "Checking Dependencies" and "Checking Contributions". However, we noted that the difference (4 minutes) is not significant. Although the participants who used the *OnTheme/Doc* approach had to perform additional tasks, the usage of the ontology and the proposed process may have led the participants to perform the concern identification activity in a more focused way. This may have minimized the impact on the time of execution of the *OnTheme/Doc* approach.

4.3 Hypothesis Tests

Although the values presented in Section 4.2 indicate that the usage of *OnTheme/Doc* approach provides good recall and f-Measure values with regard to CCC identification, it is necessary to perform statistical analyses by means of hypothesis tests, in order to ensure the reliability to the statements expressed in this paper. The hypotheses related to the precision metric was not tested, since the two analyzed samples did not show differences with regard to the values of this metric.

The purpose of a hypothesis test is to verify if the null hypothesis (H_0) may be rejected, with some significance level; when H_0 is rejected, the alternative hypothesis H_1 may be accepted. Before applying a hypothesis test, it is necessary to know in what type of probability distribution the data collected in the study is organized. This occurs because many hypothesis tests, such as the t-test (Montgomery, 2000), have as a prerequisite the need that data be normally distributed.

To verify if the data is normally distributed, we have applied a test known as Shapiro-Wilk test (Montgomery, 2000) and the values for recall, f-Measure and time metrics were considered normalized with a significance level of 99.9%.

To verify the hypotheses defined in Table 3, the t-test was applied. Comparing the average values for recall provided by the approaches *Theme/Doc* (average = 40.81) and *OnTheme/Doc* (average = 73.46), the H_{0Re} null hypothesis can be rejected with significance level of 99.9% (*p-value = 0.0004*). This means that, with 99.9% of confidence, we can say that the recall provided by *OnTheme/Doc* approach is higher than that one provided by *Theme/Doc*.

Similarly, comparing the average values of f-Measure metric of both approaches - *Theme/Doc* (average = 55.48) and *OnTheme/Doc* (average = 81.26) - the null hypothesis H_{0fM} can be rejected with significance level of 99.9% (*p = 0.0002*).

Regarding to the average time spent by the participants to perform the activities in the *Theme/Doc* (average = 41 min) and in the *OnTheme/Doc* (average = 45 min), it was not possible to obtain statistical evidences, with significance level equal or higher than 95%, to say that these values are different.

In summary, hypothesis tests have revealed that there are significant differences between the values for recall and f-Measure metrics measured for the two approaches in analysis, and the *OnTheme/Doc* approach presented better results. However, it is not possible to say that there are significant differences between the values for precision provided by both approaches, as well as for the time required to perform their activities.

4.4 Threats to Validity

Wohlin et al., (2012) state that an experimental study may face situations that threaten the validity of its results. The main threats addressed in this study are:

1) **Conclusion Validity.** This kind of threat refers to issues that affect the ability to draw correct conclusions about the experimental results. An example of this kind of threat is the choice of appropriate statistical methods for data analysis. In the case of this study, one of the statistical tests used was the t-test, which requires normally distributed data. To verify the normality of the data and minimize this threat, the Shapiro-Wilk test was applied and the result was positive for the samples.
2) **Internal Validity.** It refers to issues that may affect the ability to ensure that the results were, in fact, obtained from the treatments (*i.e.* the AORE approaches: *OnTheme/Doc* and *Theme/Doc*) and not by coincidence. A threat of this kind can be related to the strategy used to select and group the participants of the experimental study. To mitigate this threat, we did not demonstrate expectations for any approach during the training phase. In addition, the participants were grouped according to their levels of experience.
3) **External Validity.** This kind of threat refers to issues that affect the ability to generalize the results of an experiment to a wider context. In this case, the relevant factors that could have influenced the results of this study are: (i) the application used in the study, *i.e.*, *Health Watcher*; (ii) the quality of the resources (the CCC ontology and the requirements document) presented to the participants; (iii) the amount of participants of the study; and (iv) the use of undergraduate and graduate students in Computer Science. In order to mitigate these potential threats, we intend to replicate this experiment with other groups of participants and different applications.

5 RELATED WORK

Several AORE approaches have been proposed in last years; among them, many approaches address the concern identification and classification activity. In a systematic mapping (SM), conducted by the authors of this work (Parreira Júnior and Penteado, 2014), it was noted that until 2013, there were thirty-eight different AORE approaches and twenty-two of them were related to this activity. Among these, ten provided resources to support software engineers during this activity (Agostinho et al., 2008; Sampaio et al., 2005; Chitchyan et al., 2006; Zheng et al., 2010; Liu et al., 2009; Soeiro et al., 2006; Moreira et al., 2005; Chernak, 2012; Whitlle and Araújo, 2004; Alencar et al., 2010; Brito and Moreira, 2003; Mussbacher et al., 2010).

These approaches aimed to support the software engineer during the concern identification and classification through guidelines, such as catalogues of Non-Functional Requirements (NFR) (Chung and Leite, 2000; Cysneiros, 2014) or catalogues of CCC that were extensions of NFR catalogs (Moreira et al., 2005). Some problems with regard to the usage of these catalogs are (López et al., 2008): (i) they are complex to be read and understood by humans; and (ii) they are not prepared for automated semantic processing. In addition, most of these approaches does not present a process that helps the software engineer on how to use the guidelines.

Another problem that was noted from the systematic mapping is that only five of these ten AORE approaches (Sampaio et al., 2005; Soeiro et al., 2006; Moreira et al., 2005; Chernak, 2012; Brito and Moreira, 2003) were evaluated with some kind of experimental study. Hence, there is no way of knowing on the effectiveness of these approaches.

In another recent SM conducted by the authors of this study, it was found that several ontology-based approaches have been proposed for the requirements engineering field, however, none of them is specific to the context of AORE. Maybe, one of the closest works, related to this paper, is that one proposed by López et al., (2008). In this work, the authors presented an ontology for sharing and reusing NFR and design decisions. The proposed ontology aims to store the knowledge related to the NFR and design decisions, based on the description of NFR catalogues. The researcher can create instances, from this ontology, that address the NFR and design decisions of interest.

The proposal of López et al., (2008) differs from that one proposed in this paper as following: (i) their work is not related to the AORE field, therefore, it

does not address specific features of CCC, such as the classification of a CCC as non-functional or functional one, the relationships between CCC and keywords, the decomposition of concerns into sub-concerns, among others; (ii) their work does not present a process or a set of guidelines that helps the software engineer on how to use the proposed ontology; and (iii) the work does not present any kind of an experimental study on the proposal.

6 FINAL REMARKS

Based on the problems reported in recent work (Herrera et al., 2012; Sampaio et al., 2007), and already mentioned in this paper, it is possible to note that the concern identification and classification activity from requirements documents is a relevant and challenging research subject yet.

This paper presented an extension of a well-known AORE approach (*Theme/Doc*), called *OnTheme/Doc*; the aim of this extension is to improve the values for recall and precision with regard to the concern identification and classification. The main innovation of *OnTheme/Doc* approach is the usage of ontologies (*OntoCCC*) to support the users during the CCC identification. An experimental study conducted on *OnTheme/Doc* showed that the usage of ontologies may improve the values for recall, without negatively impact on the execution time and precision of the approach.

As future work proposals, we intend to: (i) register other kinds of concerns as instances of the *OntoCCC* ontology; (ii) create a computational tool for concern identification, based on instances of the *OntoCCC* ontology; and (iii) extend the *OntoCCC* ontology to include the concepts and relationship of base concerns (non-crosscutting concerns).

REFERENCES

Agostinho, S. et al. A Metadata-driven approach for aspectoriented requirements analysis. In: *10th Int. Conf. on Enterprise Inform. Syst. Barcelona*, Spain, p. 129-136, 2008.

Alencar, F. et al. Towards modular i* models. In: *ACM Symposium on Applied Computing*, p. 292-297, 2010.

Ali, B. S.; Kasirun, Z. M. D. An approach for crosscutting concern identification at requirements level using NLP. *Int. Journal of Physical Sciences*, v. 6(11), p. 2718-2730, 2011.

Araújo, J.; Whittle, J.; Kim, D. K. Modeling and composing scenario-based requirements with aspects. In: *Requirements Engineering Conference*. Washington, USA, 2004.

Baniassad, E.; Clarke, S. Theme: An approach for aspect-oriented analysis and design. *26th Int. Conf. on Soft. Eng. USA*, 2004.

Brito, I.; Moreira A. Towards a Composition Process for Aspect-Oriented Requirements. *EA Workshop*. Boston, USA, 2003.

Chernak, Y. Requirements Composition Table Explained. In: *20th IEEE Int. Requirements Engineering Conference*. Chicago, Illinois, USA, pp. 273-278, 2012.

Chitchyan, R.; Sampaio, A.; Rashid, A.; Rayson, P. A tool suite for aspect-oriented requirements engineering. In: *Int. Workshop on Early Aspects at ICSE*. p. 19-26, 2006.

Chitchyan, R. et al. Report synthesizing state-of-the-art in aspect-oriented requirements engineering, architectures and design. *Technical Report*. Lancaster University. 259 p., 2005.

Chung, L.; Leite, J. S. P. Non-Functional Requirements in Software Engineering: *Springer*, 441 p., 2000.

Clarke, S.; Baniassad, E. Aspect-Oriented Analysis and Design: The Theme Approach. *Addison-Wesley*, 2005.

Cysneiros, L. M. Catalogues on Non-Functional Requirements. Available at: http://www.math.yorku.ca/~cysneiro/nfrs/nfrs.htm. Last access: Nov. 2014.

Dijkstra, E. W. A Discipline of Programming. *Pearson Prentice Hall*, 217 p., 1976.

Falbo, R. A. et al. Um Processo de Engenharia de Requisitos Baseado em Reutilização de Ontologias e Padrões de Análise. In: *Jornada Iberoamericana de Eng. del Soft. e Engeniería del Conocimiento*, Lima, Perú, 2007 (in Portuguese).

Fensel, D. Ontologies: silver bullet for knowledge management and electronic commerce. *Springer-Verlag*. 138 p., 2001.

Gruber, T. R. Towards Principles for the Design of Ontologies used for Knowledge Sharing. In: *Int. Journal of Human-Computer Studies*, v. 43(5-6), p. 907-928, 1995.

Grundy, J. Aspect-Oriented Requirements Engineering for Component-based Software Systems. In: *4th IEEE Int. Symp. on Requirements Eng. Limerick*, Ireland, p. 84-91, 1999.

Guarino, N. Formal Ontology in Information System. In: *First Int. Conf. on Formal Ontology in Inf. Sys. Italy*, p.3-15, 1998.

Health Watcher. Available at: http://www.cin.ufpe.br/~scbs/testbed/requirements/aore/. Last access: Nov. 2014.

Hernandes, E. C. M. Um processo automatizado para tratamento de dados e conceituação de ontologias com o apoio de visualização. *Master dissert. UFSCar*, 2009 (in Portuguese).

Herrera, J. et al. Revealing CCC in Textual Requirements Documents: An Exploratory Study with Industry Systems. In: *Braz. Sym. on Soft. Eng. Natal*, BR, 2012.

Horrige, M. et al. A Practical Guide to Building OWL Ontologies Using Protégé 4 and CO-ODE Tools.

Tutorial. Univ. of Manchester, 2011.

Kit, L.K.; Man, C.K.; Baniassad, E. Isolating and relating concerns in requirements using latent semantic analysis. In: *ACM SIGPLAN Notices*, v. 41(10), p. 383-396, 2006.

Lima, J. C.; Carvalho, C. L. Ontologias - OWL. *Tech. Report*. Federal Univ. of Goiás, Brazil. 2005.

Liu, X.; Liu, S.; Zheng, X. Adapting the NFR framework to aspectual use-case driven approach. In: Int. Conf. on Soft. Eng. Research, *Manag. and Applic. Hainan Isl.*, China, 2009.

López, C., Cysneiros, L. M., and Astudillo, H. NDR Ontology: Sharing and Reusing NFR and Design Rationale Knowledge. In: *Int. Work. on Managing Req. Knowl. USA*, p. 1-10, 2008.

Montgomery, D. C. Design and Analysis of Experiments, 5ª ed., *Wiley,* 2000.

Moreira, A.; Rashid, A.; Araújo, J. Multi-dimensional Separation of Concerns in Requirements Engineering. In: *13th Int. Conf. on Req. Eng. Paris*, France, p. 285-296, 2005.

Mussbacher, G.; Amyot, D.; Araújo, J.; Moreira, A. Requirements modeling with the aspect-oriented user requirements notation (AoURN): A case study. In: *Transactions on Aspect-Oriented Software Development*. Springer-Verlag, Berlin, Heidelberg, p. 23-68, 2010.

Parreira Júnior, P. A.; Penteado, R. A. D. Aspect-Oriented Requirements Engineering: A Systematic Mapping. In: *XVI International Conference on Enterprise Information Systems*, 2014, Lisboa/Portugal, 2014.

Penim, A.S.; Araújo, J. Identifying and modeling aspectual scenarios with theme and MATA. In: *ACM Symposium on Applied Computing*. Switzerland, p. 287-291, 2010.

Rashid, A.; Moreira, A.; Araújo, J. Modularisation and composition of aspectual requirements. In: *2nd Int. Conf. on Aspect-Oriented Soft. Development*. New York, USA, 2003.

Sampaio, A.; Greenwood P.; Garcia, A. F.; Rashid, A. A Comparative Study of Aspect-Oriented Requirements Engineering Approaches. In: *Int. Symp. on Empirical Soft. Eng. and Measurement*. Madrid, Spain, p. 166-175, 2007.

Sampaio, A.; Chitchyan, R.; Rashid, A.; Rayson, P. EA-Miner: a Tool for Automating Aspect-Oriented Requirements Identification. In: *Int. Conf. Automated Soft. Eng*. California, USA, p. 353-355, 2005.

Soeiro E.; Brito, I. S; Moreira, A. An XML-Based Language for Specification and Composition of Aspectual Concerns. In: *8th Int. Conf. on Enterprise Inf. Sys*. Paphos, Cyprus, 2006.

Whittle J.; Araújo, J. Scenario Modeling with Aspects. In: *IEEE Software*, v. 151(4), p. 157-172, 2004.

Wohlin, C. et al. Experimentation in Software Engineering: an introduction. *Springer*. 249 p, 2012.

Zheng, X.; Liu, X.; Liu, S. Use case and non-functional scenario template-based approach to identify aspects. *2nd Int. Conf. on Comp. Eng. and Applications*. Indonesia, p. 89-93, 2010.