Journal of Fundamental and Applied Sciences

ISSN 1112-9867

Available online at http://www.jfas.info

Research Article

Special Issue

PROPOSITION OF IM-DECRUD TOOL TO SUPPORT ENGINEERING TASKS FOR REQUIREMENTS AND DESIGN CROSSCUTTING CONCERNS

J. Jasmis[1,*], S. J. Elias[2], R. Abd Razak[1] and W. F. Abbas[3]

[1]Faculty of Computer and Mathematical Sciences, Universiti Teknologi MARA, Jasin Campus, 77300 Merlimau, Jasin, Melaka, Malaysia

[2]Faculty of Computer and Mathematical Sciences, Universiti Teknologi MARA, 08400 Merbok, Kedah, Malaysia

[3]Faculty of Computer and Mathematical Sciences, Universiti Teknologi MARA, 40450 Shah Alam, Selangor, Malaysia

Published online: 30 May 2018

ABSTRACT

This study offered a tailored-design, prototype and produced tool as a proof-of-concept of the proposed IM-DeCRuD approach. Main features of the identified IM-DeCRUD prototype are: requirements specification definition, requirements specification modification, requirements prioritization setting and graphics visualizing representation which were produced by using the Generic Modelling Environment (GME) case tool. Java language was applied as interpreter to incorporate the feature of the prototypes. To evaluate the pertinence of the IM-DeCRuD prototype, this study used the approach of a simple case study of a library system. As a result, during the software development and valuation activities, the prototype showed its power to simplify the tedious engineering process of requirements and design crosscutting concerns.

Keywords: identification; modularization; design composition rule and conflict dissolution (IM-DeCRuD); software design; generic modelling environment (GME).

Author Correspondence, e-mail: jamaluddinjasmis@melaka.uitm.edu.my

doi: http://dx.doi.org/10.4314/jfas.v10i2s.80

1. INTRODUCTION

The definition of software concern is, “any matter of interest in a software system”, whether it is related to a system or its environment [1]. Concerns can be either functional or non-functional. Their characteristics may vary in such a way that some are noticeable whereas some are delicate, which make them difficult to be identified. Furthermore, some of them are broadly-scoped and mostly encapsulated in various modules that are different from one another. These worries are identified to be crosscutting concerns which without a logical approach, may be very difficult to understand those requirements that are specific about software. As such, software maintenance is very challenging and its development process may lead to the unthinkable failure [2].

Crosscutting concerns in software development and maintenance are moderately becoming topical issues, when it comes to software engineering. However, current works have been able to focus on identification, modularization, composition and conflict analysis of crosscutting concerns at requirements level. This is understandable since requirements documentations are mostly related to a specific high-level language concerns [3]. Nonetheless, these works did not successfully specify crosscutting properties for functional and non-functional concerns in both requirements and design phases. Furthermore, it is clearly noted that absence of proper approaches, capable of defining or state the appropriate features of crosscutting concerns to a diversified level of phases in software development activities. This has been able to make a sufficient guideline for software engineers impossible, as to the development phases of crosscutting concerns across. [4].

Mining process could be involved, in a large volume of high level specification, document, when crosscutting concern is involved. In the most serious case, documents such as interview transcripts are mostly lack accuracy and are unclear. Furthermore crosscutting concerns are often spread across document that makes it difficult to identify them [1, 5]. Based on these reasons, an automated tool is needed to minimize human error and user intervention when engineering tasks are performed.

This paper focuses on the presentation of the preliminary version of IM-DeCRuD tool as a proof-of-concept, that supports our Identification, Modularization, Design Composition

Rule and Conflict Dissolution (IM-DeCRuD) approach [6-7]. The objective of IM-DeCRuD tool is to facilitate a better understanding and reasoning to ease engineering tasks towards crosscutting concerns. Additionally, it is meant to suit the evolution process of crosscutting concerns between the requirements and their dependencies during the design phase. In specific, IM-DeCRUD tool was introduced to accommodate speedy changes of requirements for different sizes of software development in addition to maintenance projects. This paper is organized as follows: Section 2 presents the overview of the IM-DeCRUD approach, follow by Section 3 that presents the tool, which includes the contextual background of the architecture and features of the prototype tool. In addition, Section 4 discusses a case study of a simple library system that is applied. Finally, Section 5 concludes the paper and highlights the reasons for future research.

2. IM-DECRUD APPROACH: AN OVERVIEW

Figure 1 depicts an overview of the conceptual framework for Identification, Modularization, Design Composition Rule and Conflict Dissolution (IM-DeCRuD). The approach is divided into three main task; Identification and Specification, Composition and Conflict Handling which can be accomplished iteratively and incrementally. More detailed description of these tasks will be discussed in the following subsections.

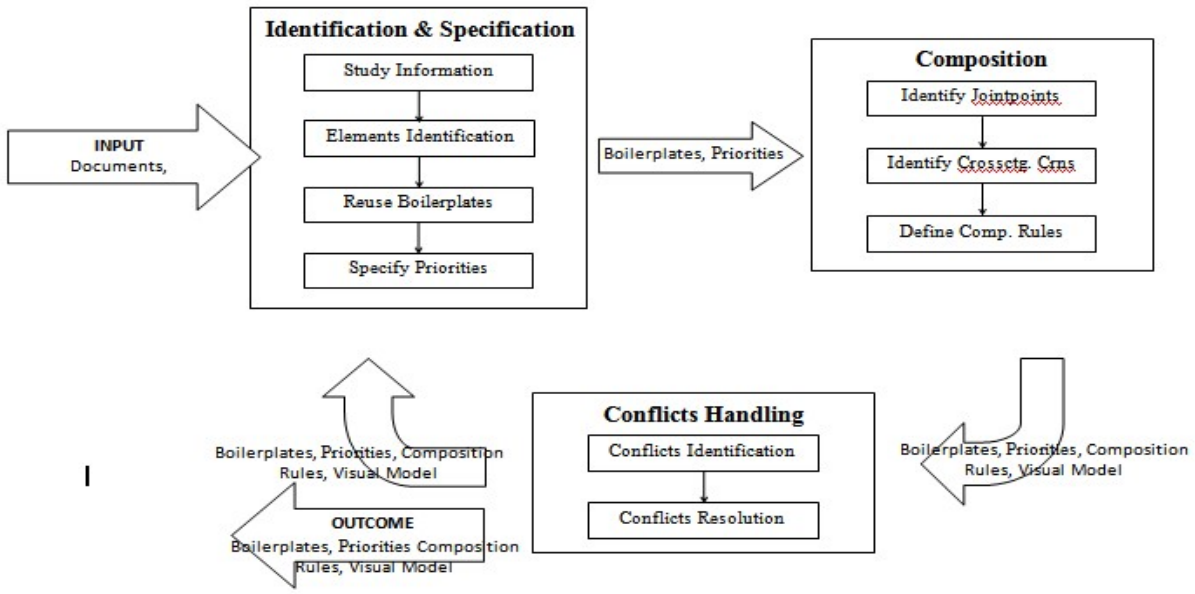

Fig.1. Conceptual framework of IM-DeCRuD approach

2.1. Identification and Specification

Identification and Specification were the starting point of the approach that contains compiling and reviewing a high level requirements documentations, followed by the extraction process upon the availability of nouns, verbs and systems' properties from each requirement viewpoints, FURs (functional concerns) and Product-Oriented NFURs (non-functional concerns) by the domain experts respectively. In addition, the process of compiling, organizing and recording those requirements components from multiple sources is accomplished by applying a specific-purpose boilerplates. This subtask is fulfilled by specifying priorities of the identified NFURs to determine the stakeholders, with the value taken as 'High', or 'Low' to identify conflict purposes that will be carried out later in other subtitles.

2.2. Composition

The Composition was carried out with the probability of weaving together the identified requirements components. With this specific purpose, our proposed composition rule, in which the structure is governed by the XML schema with its operators adopted from LOTOS (Language of Temporal Ordering Specifications) [8] is applied. This composition rule functioned as a meta-language to assist the formation of graphical notation as an alternative view for standard software high level design since the latter is incapable to specify crosscutting concerns [1, 2, 9].

2.3. Conflict Handling

Conflicts Handling dealt with the identification task with regards to conflicting crosscutting concerns, (NFURs) that were similar or came under a shared category even though they have common priorities and conflicting quantification measurement that constrain common conventional concern (FUR)., These conflicting crosscutting concerns issues were conveyed and compromised (dissolution) with the stakeholders. Details descriptions on this approach were also mentioned in [6, 7].

3. ARCHITECTURE OF IM-DeCRuD PROTOTYPE

The IM-DeCRuD To facilitate the three specific jobs, prototype tool was developed through Java language The three specific requirements are: 1. boilerplates entries management, ii requirements boilerplates creation and modification and iii propagating stored requirements components to high level design artifacts of software. The tool also integrates four features, such as: requirements specification definition, requirements specification modification, requirements prioritization setting and graphics visualizing representation.

IM-DeCRuD tool output, can be exported to the domain-specific design modeling environment case tool; GME [10]. GME modeling, is an open source tool that can be used for any specific domain application that is based on the meta-modeling approach. Using this approach means that, the theoretical specification of requirements and design crosscutting concerns in the conceptual modeling was formalized. Figure 2 depicts the architecture of the IM-DeCRuD prototype environment in general which was formulated of the features mentioned above. Meanwhile, Figure 3 shows the main visual of IM-DeCRuD prototype, where the features of the prototype are described in the following subsections.

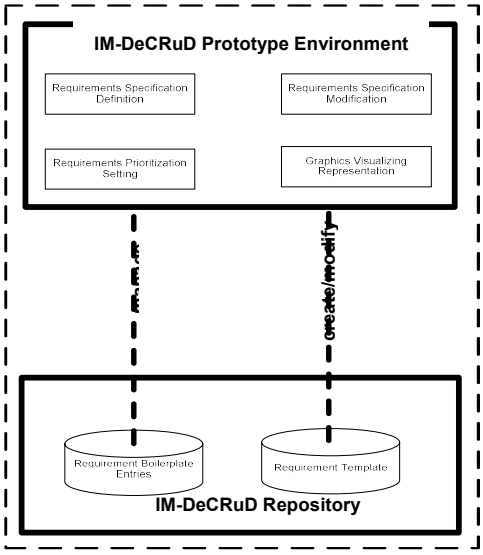

Fig.2. Architecture of IM-DeCRuD Approach

3.1. Requirements Specification Definition

The Requirements Specification Definition feature allows the engineers to transcribe the requirements specifications that were obtained from various high level sources through the requirements templates metamodel. Our previous works suggested four categories of requirements such as system functionality, performance, quality and control.

In addition, Figure 3 also illustrates an instance of the prototype view, where the engineer was given a chance to choose any of the categorical requirements mentioned above and its respective templates, as well as accomplishing those related placeholders, (based on the requirements components and quantitative measurements) for each of the requirements specification. Because the construction of a complex software system is usually derived from many partial or redundant requirements documentations to reflect different perspectives of end users [11], there will be a usual requirements specifications of many lines from the requirements statements which were recognized by their source of origin.

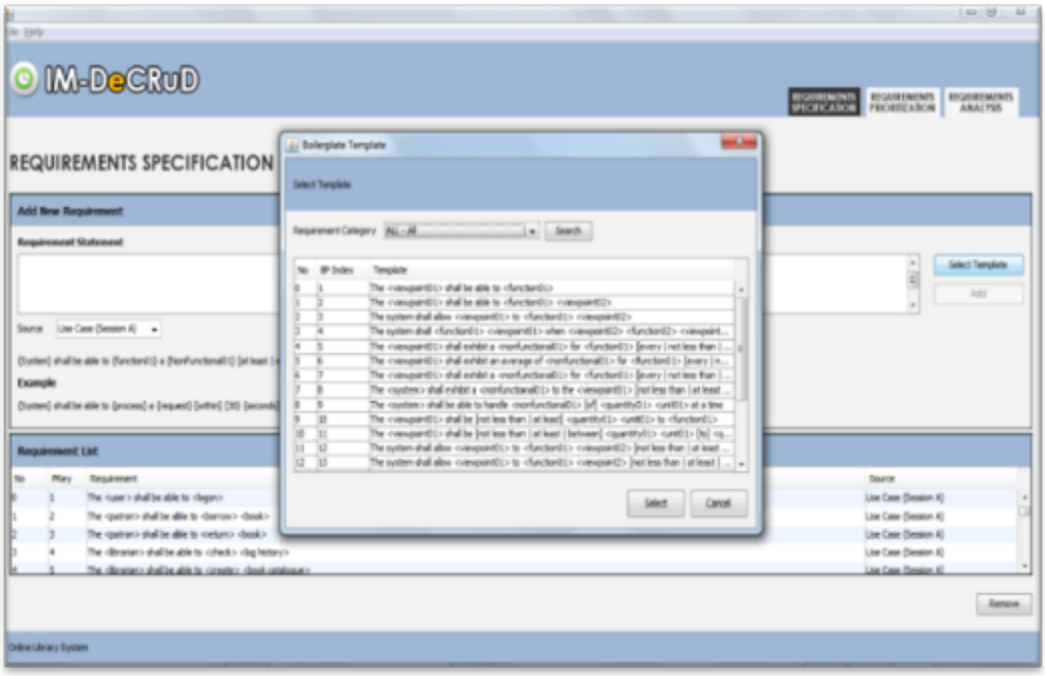

Fig.3. IM-DeCRuD prototype environment of defining new requirement

3.2. Requirements Specification Modification

The engineer is allowed to do plus or minus to the desired requirements specifications. The prototype will then automatically propagate the impact on the respective software design elements.

3.3. Requirements Prioritization Setting

This feature allows a specific NFUR component according to its priority level that the stakeholders determine. This can be found in Figure 4, where every NFUR component was set to “High” or “Low” before those settings were saved.

3.4. Graphics Visualizing Representation

This feature visualized the graphical representation (obtained from the automatically generated composition rules in XML format) for the correlations between FURs (jointpoints) and its associated viewpoints along with the NFURs as shown in Figure 5. Each line connected to the conflicted NFUR component was stipulated and labeled, according to its level of priority (indication for attention). Besides, the use of GME toolkit, make the composition rule (XML format) became readable and viewed in a software high level design (class diagram) as a substitution view for the graphical representation.

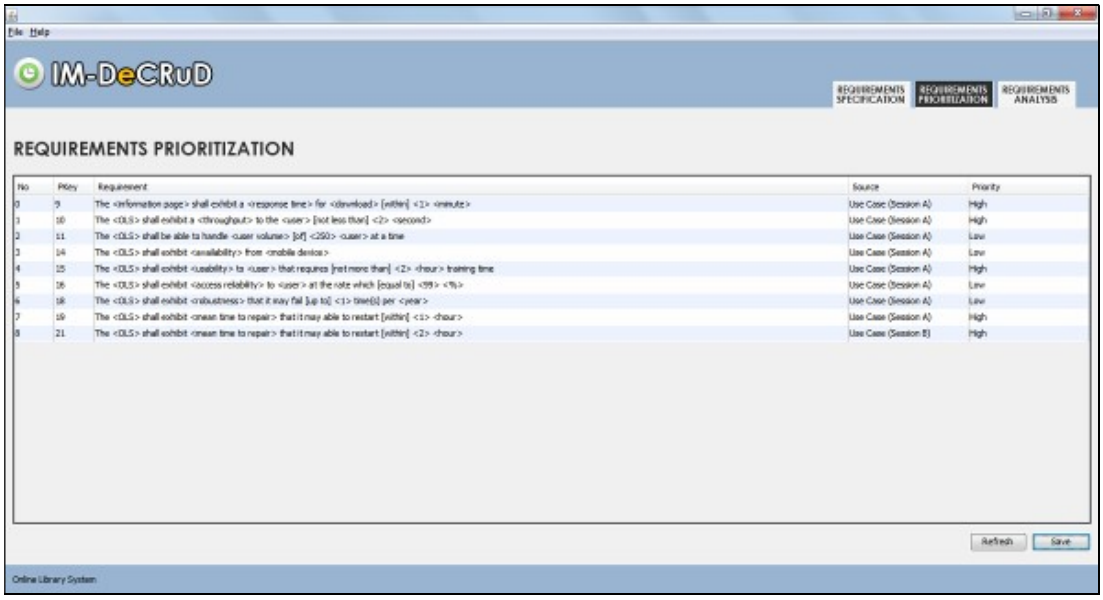

Fig.4. IM-DeCRuD's requirements prioritization setting view

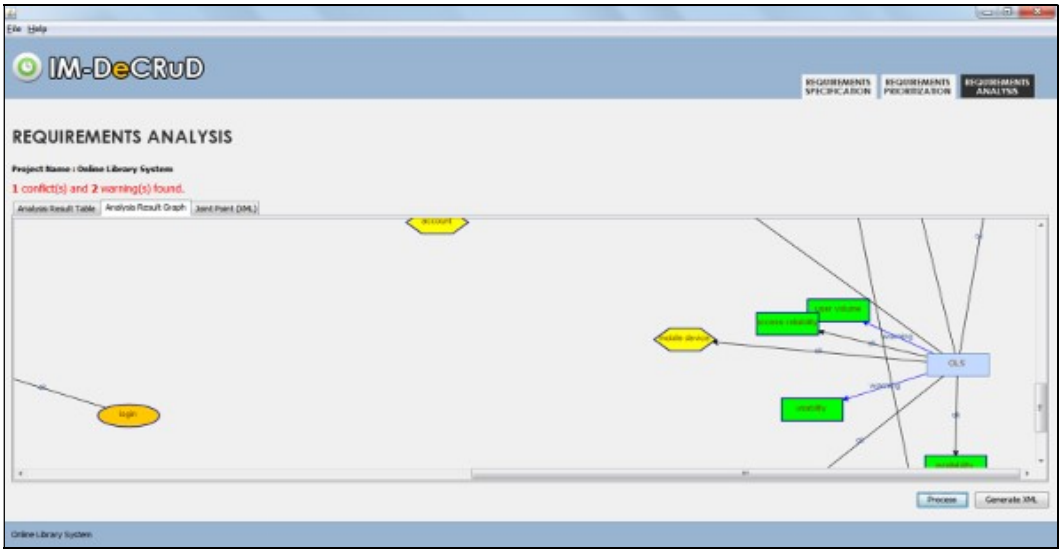

Fig.5. NFUR-FUR-viewpoints relationships

4. LIBRARY SYSTEM CASE STUDY EVALUATION

The objective of using this simple library system case study application and evaluation was to get the first suggestion upon the implementation of IM-DeCRuD approach in a factual environment of software development life cycles before the study continue to apply this to the real industry-related case study. The appropriate requirements for the library system domain were specified, based on the template of requirements definition features as provided by the prototype. The associated software design artifact such as the

conceptual graphical representation and class diagram was created for all the related requirements specifications. Upon that, the following subsections explain how the library system functions.

4.1. Step 1: Identify and Define Requirements Specifications of Library System

The case study was designed and conducted in two separate sessions that involves different levels of stakeholders, hosted by the requirement engineers. An example of a used case for the library system can be seen in Figure 6. Requirement engineers reviewed all the information and particulars captured in the form of a used cases diagrams and documentations, that designate the same system. After that, the requirements components were distinguished and extracted. In addition, by using the prototype, the engineers decided on appropriate requirements templates based on their categories and the accessible placeholders "< >" were filled with suitable keywords according to the previously identified requirements components. In the meantime, the placeholder [| |] functioned as equality and relational operators to the quantification attribute of the placeholders for the NFURs and viewpoints. The source of each line of the requirements specifications was then specified according to the source of session captured requirements Some example of requirements specifications of library system were shown in Table 1. This was specified according to the requirements template definitions (the source from which session is not shown).

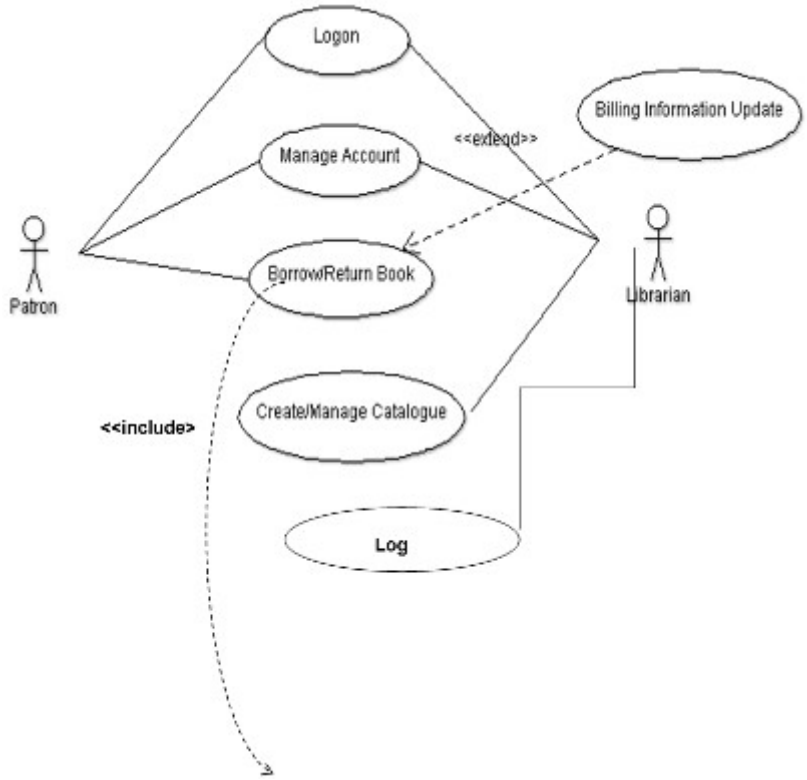

Fig.6. Use case diagram of on-line library system

Table 1. Requirements template definition

Category of Requirements	Requirements Specification
System Functionality	The <**user**> shall be able to <**logon**>
	The <**patron**> shall be able to <**borrow**> <**book**>
	The <**patron**> shall be able to <**return**> <**book**>
	The <**librarian**> shall be able to <**check**> <**log history**>
	The <**librarian**> shall be able to <**create**> <**book catalogue**>
	The <**librarian**> shall be able to <**manage**> <**book catalogue**>
	The <**user**> shall be able to <**manage**> <**account**>
	The system shall <**update**> <**billing information**> when <**patron**> <**borrow**> <**book**>
Performance	The <**information page**> shall exhibit a <**response time**> for <**download**> [**within**] <**1**> <**minute**>
	The <**OLS**> shall exhibit a <**throughput**> to the <**user**> [**not less than**] <**2**> <**second**>
Quality	The <**password**> shall be [**at least**] <**8**> <**character**> to <**login**>
	The system shall allow <**patron**> to <**borrow**> <**book**> [**at minimum period of**] <**1**> <**day**>
	The <**OLS**> shall exhibit <**availability**> from <**mobile device**>
	The <**OLS**> shall exhibit <**usability**> to <**user**> that requires [**not more than**] <**2**> <**hour**> training time
	The <**OLS**> shall exhibit <**access reliability**> to <**user**> at the rate which [**equal to**] <**99**> <**%**>
	The <**OLS**> shall exhibit <**robustness**> to <**user**> that it may fail [**up to**] <**1**> time(s) per <**year**>
	The <**OLS**> shall exhibit <**mean time to repair**> that it may able to restart [**within**] <**1**> <**hour**>
Control	The system shall not allow <**unauthorized user**> to <**logon**>

4.2. Step 2: Specify Requirements Priority

Each NFUR component was specifically priority according its level of 'High' or 'Low' as determined by the stakeholders and it was also pertinent to the common requirements specifications from different sources.

4.3. Step 3: Analyze Requirements Specification

Next, some conflicting NFUR's requirement specifications were listed as "Conflict" or "Warning" based on their priority level in addition to their quantification features that were determined by the specific-purpose conflict identification procedure as mentioned in [6]. For the time being, there were some cases that were in the alert list, where two or more common requirements specifications, whose viewpoint components contain conflicting values on quantification feature. Figure 7 illustrates the "Conflict" and "Warning" lists for NFURs components (however, in this case, the alert list for viewpoint components are not obtainable.).

A directed graph established by requirements components (viewpoints, NFURs) were affixed by a jointpoint (FUR). It is functional in such that, these conflicted NFUR lines affixed to a common FUR component were highlighted and labeled according to the type of conflicts. These may lead to a revision of the requirements specification among the engineers and the stakeholders.

Fig.7. "Conflict" and "Warning" Lists for NFUR components

Besides that, the directed graph composition rules were also produced and saved to a .xme file to be viewed in class diagram format by using the GME toolkit as shown in Figure 8.

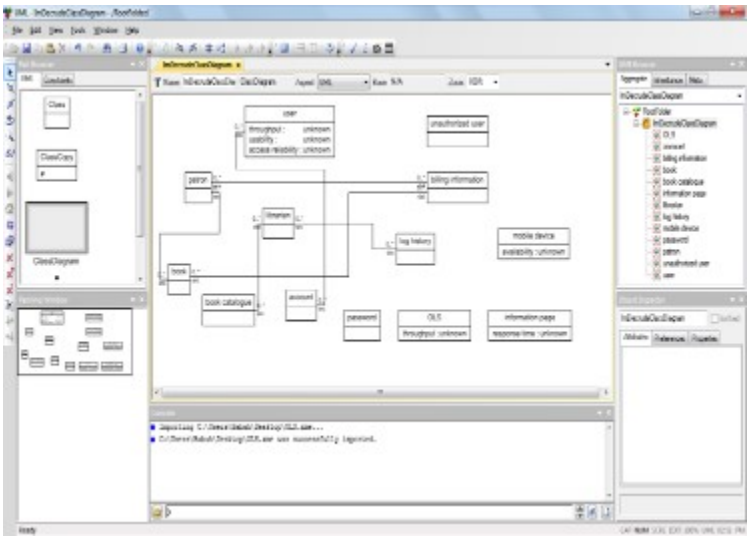

Fig.8. Overall class diagram of on-line library system in GME environment

5. RESULTS AND DISCUSSION

Firstly, the application of a library system case study that uses IM-DeCRuD showed that, the prototype is capable of dealing with crosscutting concerns among different types and levels of software artifacts in software development and evolution activities.

Secondly, the tailored-designed prototype was found to be relevant and useful for software engineers to transcribe requirements specifications, in a standardized manner suitable to the predefined templates. The category that classifies the requirements templates (functionality, performance, quality and control) embodied in the prototype was capable to simplify the transcribing process in terms of selecting the most appropriate template for every requirement specification.

The main objective was attained through, the tool that has shown its automatic support in managing and simplifying the evolution process towards crosscutting concerns from requirements level to the design level. In addition to that, the automated support does not only helped to lessen engineers' human effort, but also assisted in reducing human errors to determine the impacted element in software artifacts simultaneously.

6. CONCLUSION AND FUTURE WORK

The development of the IM-DeCRuD prototype conferred in this paper was developed as proof-of-concept to support the crosscutting concerns evolution process. In order to evaluate the applicability of the prototype, the application of a simple library system case study was used. The preliminary findings from the evaluation results demonstrated that, the IM-DeCRuD prototype is pertinent to make the tedious and erroneous engineering process more comprehensible between requirements and design phases (class diagrams). As a next research, we plan to apply a real world industry-related case study, to be tested against the prototype. Future recommendations of the study also plan to improve the graphical representation of the requirement components to have a scalable view so that it can reflect the different sizes of software development as well as maintenance projects.

REFERENCES

[1] Ali B S, Kasirun Z M. Developing tool for crosscutting concern identification using NLP. In IEEE International Symposium on Information Technology, 2008, pp. 1-8

[2] Ali B S, Kasirun Z M. A review on approaches for identifying crosscutting concerns. In IEEE International Conference on Advanced Computer Theory and Engineering, 2008, pp. 855-859

[3] Rashid A, Moreira A, Araújo J. Modularisation and composition of aspectual requirements. In ACM 2nd International Conference on Aspect-Oriented Software Development, 2003, pp. 11-20

[4] Sanchez P, Fuentes L, Jackson A, Clarke S. Aspects at the right time. Tiergartenstrasse 17, Heidelberg, D-69121, Germany, 2007, pp. 54-113

[5] Sampaio A, Rashid A, Rayson P. Early-aim: An approach for identifying aspects in requirements. In 13th IEEE International Conference on Requirements Engineering, 2005, pp. 487-488

[6] Jasmis J, Ibrahim S, Anom R B, Elias S J. IM-DeCRUD: An approach for requirements and design crosscutting concerns to support software evolution. In Postgraduate Annual Research on Informatics Seminar, 2012

[7] Jasmis J, Ibrahim S, Anom R B, Elias S J. Proposition on criteria and approach for

requirements and design crosscutting concerns to support software evolution. International Journal of Research and Reviews in Computer Science, 2011, 2(3):725-730

[8] Marsan M A, Bianco A, Ciminiera L, Sisto R, Valenzano A. A LOTOS extension for the performance analysis of distributed systems. IEEE/ACM Transactions on Networking, 1994, 2(2):151-165

[9] Brito I. Aspect-Oriented requirements engineering. In 7th International Conference on Unified Modelling Language, 2004, pp. 1-8

[10] Davis J. GME: The generic modeling environment. In 18th Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications, 2003, pp. 82-83

[11] Spanoudakis G, Finkelstein A. Reconciling requirements: A method for managing interference, inconsistency and conflict. Annals of Software Engineering, 1997, 3(1):433-457

How to cite this article:
Jasmis J, Elias S J, Abd Razak R, Abbas W F. Proposition of im-decrud tool to support engineering tasks for requirements and design crosscutting concerns. J. Fundam. Appl. Sci., 2018, *10(2S), 1080-1093.*