IM-DeCRUD Prototype: A Tool to Support Engineering Tasks for Requirements and Design Crosscutting Concerns

Jamaluddin Jasmins, Suhaimi Ibrahim, Raja Baharuddin Anom, Ahmad Efendi Mohd Hassan, Shamsul Jamel Elias

Identification, Modularization, Design Composition Rule and Conflict Dissolution (IM-DeCRU-D) approach has been proposed previously in order to promote a simple but significant way to support rapid changes of crosscutting concerns at requirements and design phases for various sizes of software development and maintenance projects. In this research, a tailored-design prototype tool has been developed as a proof-of-concept of the proposed IM-DeCRU-D approach. Four main features of the identified IM-DeCRUD prototype are: requirements specification definition, requirements specification modification, requirements prioritization setting and graphics visualizing representation. These components have been developed using Generic Modeling Environment (GME) case tool and Java language is used as an interpreter to incorporate the prototype features. Simple case study of a library system is applied to assess the applicability of the IM-DeCRU-D prototype. As a result, the prototype has shown its ability to simplify the tedious of engineering process of requirements and design crosscutting concerns during the software development and evolution activities.

Keywords: Identification, Modularization, Design Composition Rule and Conflict Dissolution (IM-DeCRU-D); software design; Generic Modeling Environment (GME)

1. INTRODUCTION

Software concern are defined as “any matters of interest in a software system”, whether be related to a system or its environment [1]. Concerns can be functional or non-functional. Their characteristic may be varies in such some are obvious and some are subtle which make it difficult to identify them. Moreover, some of them are broadly-scoped and usually encapsulated in several different modules. These concerns are identified as crosscutting concerns and without systematic approach may cause particular software requirements become very hard to understand. Software are difficult to be maintained and evolution process will lead to failure per se [2].

Crosscutting concerns in software development and maintenance have gradually become topical issues in software engineering. However, recent works are focusing on identification, modularization, composition and conflict analysis of crosscutting concerns at requirements level. It is justifiable since requirements documentations are always related to high-level language to specify concerns [3]. Nevertheless, these works fail to effectively specify crosscutting properties for functional and non-functional concerns at both requirements and design phases. Furthermore, it is noted that there are lack of approaches that are able to define appropriate characteristics of crosscutting concerns for diversified level of phases in software development activities. As such, this leads to the inability to provide sufficient guidelines for software engineers to attend to crosscutting concerns across development phases [4].

Dealing with crosscutting concerns may involves mining process towards large volume of high level specification documents. Worse case, documents such as interview transcripts are usually lack of accuracy and vague. Furthermore crosscutting concerns are often scattered across documents which cause their identification difficult [1, 5]. For such reason, an automated tool seems desired to minimize human error and user intervention while performing the engineering job.

The main focus of this paper is to present the preliminary version of IM-DeCRU-D tool as a proof-of-concept that supports our Identification, Modularization, Design Composition Rule and Conflict Dissolution (IM-DeCRU-D) approach [6, 7]. The objective of IM-DeCRU-D tool is to facilitate better understanding and reasoning for engineering tasks towards crosscutting concerns. Moreover, it is aimed to suit with evolution process of crosscutting concerns between requirements and its dependencies at design phase. Specifically, IM-DeCRU-D tool is introduced to cater for rapid changes of requirements for various sizes of software development as well as maintenance projects. The remaining work is organized as follows. Section 2 presents some overview of the IM-DeCRU-D approach. Next, Section 3 presents the tool, including the contextual background of the architecture and features of the prototype tool. As a start, in Section 4, a case study of a simple library system is applied. Meanwhile, Section 5 concludes the paper and present the work plans for the future.
An overview of conceptual framework of Identification, Modularization, Design Composition Rule and Conflict Dissolution (IM-DeCRuD) approach is depicted in Figure 1. It can be divided into three main tasks: Identification and Specification, Composition and Conflict Handling and can be accomplished iteratively and incrementally. Further description of these tasks will be presented in the following subsections.

### 2.1 Identification and Specification

Identification and Specification is the starting point of the approach which involves in compiling and reviewing high level requirements documentations, followed by extraction process upon available nouns, verbs and system’s properties from each requirement for viewpoints, FURs (functional concerns) and Product-Oriented NFURs (non-functional concerns) respectively by the domain experts. Next, process of compiling, organizing and recording those requirements components from multiple sources is performed by applying specific-purpose boilerplates. This subtask is fulfilled by specifying priorities of the identified NFURs as per determined by the stakeholders, with the value taken as ‘High’, or ‘Low’ for conflicts identification purposes that will be carried out later in other subtask.

### 2.2 Composition

Composition carries out with the possibility to weave together those identified requirements components. For this specific purpose, our proposed composition rules in which the structure is governed by XML schema with its operators are adopted from LOTOS (Language of Temporal Ordering Specifications) [8] is applied. This composition rule acts as a meta-language to accommodate formation of graphical notation as an alternative views for standard software high level design since the later seems to have limitation for the purpose to specify crosscutting concerns [1, 2, 9].

### 2.3 Conflict Handling

Finally, Conflicts Handling deals with the identification task towards conflicting crosscutting concerns (NFURs) which are similar or falls under shared category besides having common priorities and conflicting quantification measurement that constraint common conventional concern (FUR). Next, these conflicting crosscutting concerns issues will be communicated and negotiated (dissolution) with the stakeholders. Detailed descriptions on this approach have been presented in [6, 7].

### 3. ARCHITECTURE OF IM-DeCRuD PROTOTYPE

The IM-DeCRuD prototype tool is developed using Java language in order to facilitate three specific jobs: i) requirements boilerplates entries management, ii) requirements boilerplates creation and modification, iii) propagating stored requirements components to software high level design artifacts. The tool also incorporates with 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 is an open source modeling tool for any specific domain application based on meta-modeling approach. By using this approach, the theoretical specification of requirements and design crosscutting concerns in the conceptual modeling can be formalized. Figure 2 illustrates the architecture of the IM-DeCRuD prototype environment in general which is composed into the abovementioned features. Meanwhile, Figure 3 portrays the main visual of IM-DeCRuD prototype in which the features of the prototype will be described in the following subsections.

![Figure 1. Conceptual Framework of IM-DeCRuD Approach](image1)

![Figure 2. Architecture of IM-DeCRuD Approach](image2)
(according to the requirements components and quantitative measurements) for each of the requirements specification. Since it is realized that the construction of a complex software system usually derived from many partial or redundant requirements documentations to reflect different perspectives of end users [11], there will be common requirements specifications at many lines of the requirements statements which are differentiated by its source of origins.

3.2 Requirements Specification Modification
This feature accommodates the engineers to add or remove the desired requirements specifications. The prototype will then propagate the impact to the respective software design elements automatically.

3.3 Requirements Prioritization Setting
This feature allows each NFUR components to be specified of its priority level as per determined by the stakeholders. As seen in Figure 4, every NFUR component is set to “High” or “Low” before those setting is saved.

3.4 Graphics Visualizing Representation
This feature visualizes the graphical representation (derived from the automatically generated composition rules in XML format) for the correlations between FURs (jointpoints) and its associated viewpoints in addition to NFURs as illustrated in Figure 5. Each line that connects conflicted NFUR component will be indicated and labeled based on its level of priority (indication for attention). Besides, by using the GME toolkit, the composition rule (XML format) would also be readable and viewed in software high level design (class diagram) as a substitution view for the graphical representation.

4. LIBRARY SYSTEM CASE STUDY EVALUATION
The objective of this simple library system case study application and evaluation is to get the first indication upon the implementation of IM-DeCRuD approach in actual environment of software development life cycles before the real industry-related case study is applied. The suitable requirements for the library system domain will be specified based on the requirements template definition features provided by the prototype. The related software design artifact such as the conceptual graphical representation and class diagram will then be created for all the related requirements specifications. As such, the following subsections will demonstrate on how the library system is implemented in the prototype.

4.1 Step 1: Identify and Define Requirements Specifications of Library System
We design the case study in which the requirements capture are conducted in two separate sessions with various levels of stakeholders hosted by the requirement engineers. Figure 6 represents an example of a use case for the library system. Requirement engineers will then study all information captured in forms of use cases diagrams and documentations that specify the same system. Then, the requirements components will be identified and extracted. Next, by using the prototype, the engineers will choose suitable requirements templates according to its categories and those available placeholders “<   >” will be filled with appropriate keywords according to the earlier identified requirements components. Meanwhile, placeholder [   |   |   ] acts as equality and relational operator to quantification attributes’ placeholders for the NFURs and viewpoints. Source of each line of those requirements specifications will then be specified in reference to which requirements capture session it was obtained.

Table 1 shows some examples of requirements specifications of library system that have been specified based on the requirements template definitions (source from which session is not shown).
### 4.2 Step 2: Specify Requirements Priority

Each NFUR component will be specified its priority of ‘High’ or ‘Low’ as per determined by the stakeholders and it is also applicable for common requirements specifications from various sources.

### 4.3 Step 3: Analyze Requirements Specification

In the next step, some conflicting NFUR’s requirement specifications are listed as “Conflict” or “Warning” according to its priority level as well as its quantification attributes, determined by our specific-purpose conflict identification procedure as in [6]. Meanwhile, there are also some cases where two or more common requirements specifications that its viewpoint components contain conflicting values on quantification attributes are being in the alert list. Figure 7 illustrates the “Conflict” and “Warning” lists for NFURs components (alert list for viewpoint components are not available in this case).

A directed graph which are formed by requirements components (viewpoints, NFURs) are connected by a jointpoint (FUR). It seems useful in such that those conflicted NFUR lines that connecting to a common FUR components are highlighted and labeled according to the types of conflicts. These may lead to a revision of the requirements specification between the engineers and the stakeholders.

### 5. DISCUSSIONS ON PRELIMINARY FINDINGS

Firstly, the application of a library system case study using IM-DeCRuD shows that the prototype has the ability to deal with crosscutting concerns among different types and level of software artifacts in software development and evolution activities.

Secondly, the tailored-design prototype is found to be applicable for the engineers in order to transcribe requirements specifications in a standardized manner according to the predefined templates. The category that classifies the requirements templates (such as functionality, performance, quality and control) that embodied in the prototype in addition, able to simply the transcribing process in term of choosing the most suitable template for each of the requirement specification.
The tool has shown its automatic support in managing and simplifying evolution process towards crosscutting concerns from requirements level to the design level by which we believe that the main target is achieved. Additionally, the automated support not only helps in reducing engineers’ human effort, but also assists to reduce human errors in determining the impacted element in software artifacts at the same time.

6. SUMMARY AND FUTURE WORK

We have presented the design and architecture of IM-DeCRuD prototype in this paper. The prototype was developed as a proof-of-concept to support crosscutting concerns evolution process. The application of a simple library system case study is used to evaluate the applicability of this prototype. The preliminary findings from the evaluation results demonstrated that IM-DeCRuD prototype is applicable enough to simplify the tedious and erroneous engineering process between requirements and design phases (class diagrams). As for future work, we will apply real world industry-related case study and test it against the prototype. We also plan to refine the graphical representation of the requirement components to have scalable view in order to reflect to various sizes of software development as well as maintenance projects.

REFERENCES


AUTHOR PROFILES

Jamaluddin Jasnis received his M.Sc. in Information Technology from Universiti Kebangsaan Malaysia (UKM), Bangi, Malaysia in 1999. From 1999 until 2004 he was serving with Universiti Tun Abdul Razak (UNITAR), Kelana Jaya, Selangor, Malaysia as a lecturer and also program coordinator. He attached to Universiti Teknologi MARA (UiTM), Shah Alam, Selangor as lecturer since 2004. He is currently pursuing his Ph.D since July 2007 with Universiti Teknologi Malaysia (UTM), Kuala Lumpur, Malaysia as a part-time and recently converted to full time student. His current research area involves requirements engineering and software evolution.

Suhaime Ibrahim is currently an Associate Professor attached to Advanced Informatics School (AIS), UTM International Campus, Kuala Lumpur, Malaysia. He received a B.Sc in Computer Science (1986) from Stratchlyde University, United Kingdom, Master in Computer Science (1990) and PhD in Computer Science (2006) from UTM. He is an ISTQB certified tester and currently being appointed a board member of the Malaysian Software Testing Board (MSTB). He has published articles in both local and international journals such as the International Journal of Web Services Practices, Journal of Computer Science, International Journal of Computational Science, Journal of Systems and Software, and Journal of Information and Software Technology. His research interests include software quality and testing, requirements engineering, software evolution, Web services and trusted computing.

Y.M. Raja Bahrudin Anom is currently a senior lecturer at Advanced Informatics School (AIS), UTM International Campus, Kuala Lumpur, Malaysia. He received both his degree and Masters in System Analysis/Business (1985) from The University of West Florida, USA. Other than teaching and research, Mr. Anom contributes many consultancy and contact research within the government and private agencies. His work contributes to the award winning prime ministers award in information technology (government agency category) in 2002. His research interests include project management information technology, information security management, systems and analysis and design.
Ahmad Efendi Mohd Hassan received his M.S. in Software Engineering from Center For Advanced Software Engineering (CASE), UTM International Campus, Kuala Lumpur, Malaysia in 1997. He joined Sapura Research Sdn Bhd as Senior Software Design Engineer in 1998 and involves with various RnD projects. He later continued his profession as Software Engineer at Sapura Energy Sdn Bhd until 2005. Due to his enormous experiences, he is currently acting as a freelancer in providing consultancy services which are majoring in Software Engineering field.

Shamsul Jamel Elias has received his M.S. in Information Technology from the University of Hull (UK) in Sept 1997. On August 2003 he has joined Universiti Teknologi MARA (Malaysia) as lecturer. His current research activities are in the field of pervasive computing, ad-hoc networking and vehicular networks.