International Journal of Software Engineering & Research Methodology – IJSERM
Volume 1 : Issue 1
Publication Date : 09 January 2014

GenSRS: Generation of Software Requirements Specification From Use case diagram and Sequence diagram

Mohammed Riyad Abdullah
Department of Software Engineering
Universiti Tun Hussein Onn Malaysia
Batu Pahat, Malaysia
hi110048@siswa.uthm.edu.m

Rosziati Ibrahim
Department of Software Engineering
Universiti Tun Hussein Onn Malaysia
Batu Pahat, Malaysia
rosziati@uthm.edu.my

***Abstract*— the reverse engineering process used to recover the requirements, and the experience gained during the study are reported. The object oriented software development is used because objects are more natural and classes of objects can be reused. The phase in OOSD is requirements analysis, design, implementation and testing. Software Requirements Specification (SRS) is a detailed description for functional and non-functional requirements of the system to be developed. GenSRS is a tool that is able to generate Software Requirements Specification (SRS) automatically according to system's requirements. In this paper, the focus is only two specific diagrams, use case diagram and sequence diagram. These diagrams are used for editing and drawing diagrams and also to check the correctness as well as consistency between diagrams. The systems' requirements are transformed using use case diagram and sequence diagram to generate SRS and save on TXT file.**

Keywords—Software Requirements Specification (SRS), Requirements Engineering (RE), Natural Language (NL), Natural Language Syntax and Semantics (NALASS).

1. Introduction

the reverse engineering process used to recover the requirements, and the experience gained during the study are reported. For example, software developers can employ reverse techniques to discover how to interoperate with undocumented or partially documented software [1]. Thus, to gather initial requirement from a particular documentation or software reverse engineering would play a very significant role. There are many Computer Aided Software Engineering (CASE) tools that help with the development of software, but they rarely provide support for the Natural Language (NL) descriptions of requirements [2]. Object-oriented Software Development (OOSD) is a methodology which views a system as collection of self-contained objects that have both data and process [3]. The object oriented software development is used because objects are more natural and classes of objects can be reused. The phase in OOSD is requirements analysis, design, implementation and testing. The requirements analysis phase of OOSD consists of activities aimed at clarifying the requirements of software system, an SRS is a document containing a complete description of a product's external behavior and corresponds to system in the "what versus how" problem just described. However, depending on how the software is being developed and who is writing the SRS, it may be detailed or general, to make this clearer [4]. The Software Requirements Specification (SRS) is a detailed description for functional and non-functional requirements of the system to be developed. The specification of software is difficult and complex activity that requires substantial effort from the requirements engineer [5]. SRS documents, which are defined in a previous research [2] as the medium used to communicate user's requirement to technical people responsible for developing the software. The Institute of Electrical and Electronic Engineers IEEE standard 830 release (1998) defined SRS as a specification for a particular software product, program, or set of programs that performs. The primary focus of existing tools lies on diagrams, charts and pictures. Less emphasis is given on the textual specification requirements [6]. This paper discusses on software requirements specification where generation shows the present of faults and proposes a generation SRS tool and reverse engineering. The rest of the paper is organized as follow. Section 2 presents the related work and section 3 discusses the system's requirements. Also present the idea on how generation SRS is done from use case diagram to sequence diagram. Section 4 discusses the used tool. Finally, a conclusion is presented in section 5 and suggestions for future work for the tool.

2. Related Work

There are several discussions on how generate SRS to help develop produce the document for software requirements. Specification of software is a difficult and complex activity that requires substantial effort on the part of requirements engineer. There are very few programs on the market that support text in the way that STORM project to create a CASE tool capable of handling the text aspects of requirements and use case modeling[7]. EventStudio is A CASE tool mainly concerned with the graphical representations of a system, it incorporates a scripting language called FDL that allows for dynamic generation of diagrams, including automatic generation of scenarios. Further, EventStudio can create

images that can be scaled to different paper sizes and can export documents. However, it does not handle text aspects of requirements, use cases and scenarios [8]. The open requirements Management tool is a powerful open source requirements tool; this tool allows for prioritization of requirements and includes a traceability matrix but lacks support for use cases, scenarios and use case diagrams [9]. Tools such as Rational Rose [10] and MagicDraw[11] provide significant capabilities for drawing diagrams and even generate code from software models but not adequate facilities for the above hence, the analysts need to write their project's SRS using regular text editors and templates, such as Microsoft Word or LaTex, and the IEEE SRS template[12]. DOORS [13] and STORM [7] these tools do not generate plain natural language descriptions such as those, for example that can be created with the use of the IEEE SRS template and they are only applicable to Use Case modeling. Their input also has to be processed first by the analyst (the analyst has to create the use case), while in NALASS the main input to the SRS document generation component, which is a set of formalized sentences, is automatically create by the tool, based on the user's answers to pre-determined questions[14].

3. The Tool

Requirements of a system are usually captured data from both diagrams and analysis phase. The functional requirements for a tool explain the functionality that a tool should provide, how a tool should respond to particular inputs, and how the proposed tool could work in a particular situations. Table 1 is the description of functional requirements.

TABLE I. TABLE TYPE STYLES

Functional Requirements		
1	**Draw and Editor Use Case Diagram and Sequence Diagram**	1. The system shall allow user to open a new project to draw Use Case Diagram and Sequence Diagram. 2. The system shall allow user to drag and drop the elements diagrams. 3. The system shall allow user to resize and delete use case diagram elements and same with sequence diagram. 4. The system shall allow user to save the diagrams. 5. The system shall allow user to write text in use case and object, message in sequence diagram.
2	**Perform Syntax Error**	1. The system shall verify the syntax errors for use case diagram or sequence diagram. 2. The system shall display error message when there are syntax errors exit in use case diagram or sequence diagram.
3	**Perform Semantic Consistency**	1. The system shall to get the details from use case to check semantic consistency. 2. The system shall allow user to view error message in separate windows.
4	**Generate SRS**	1. The system shall display error message when no data in use case. 2. The system shall allow user to save the flow of event on TXT file.

For this study, use case diagram and sequence diagram are used to specify functional requirements of the system. For the use case diagram, use cases are used to define the requirements of the system. After the drawing of the use case diagram, the user will be allowed to open the interface for each use case details to put information about the diagram then to get semantic consistency based on rules. Therefore, user can convert from each use case pattern to sequence pattern based on details in use case. The system then processes the input and produces the required output according to its requirements specification.

The tool is namely GenSRS (Generating Software Requirements Specification), it can be used to layout the use case diagram and sequence diagram of any system. Tool design is to define the interface and data for proposed system in order to satisfy specified requirements. During the tool design, a set of analysis model will be designed. As shown in the Figure 1, there is one part in the tool design which is the GenSRS tool. Only one actor interacts with tool which is user. The user connected directly with the tool through eight use cases. There is only one user interacts with the tool. User connects with tool through eight use case. The main function for the GenSRS tool is checking consistency between use case diagram and sequence diagram. User is needed to draw the use case diagram to get syntax error and semantic consistency with step flow of the event in order to make capture from use case detail. This use case is either used to draw sequence diagram. Then, the syntax errors of the whole used elements will be checked through rules. The tool will display the error message to the user in case of errors within the diagram. Also it allows the user to put in each use case details to draw sequence diagram and to show to the user the flow events in the diagram. Finally, the user is able to save the diagram in a folder and the flow of events in TXT file.

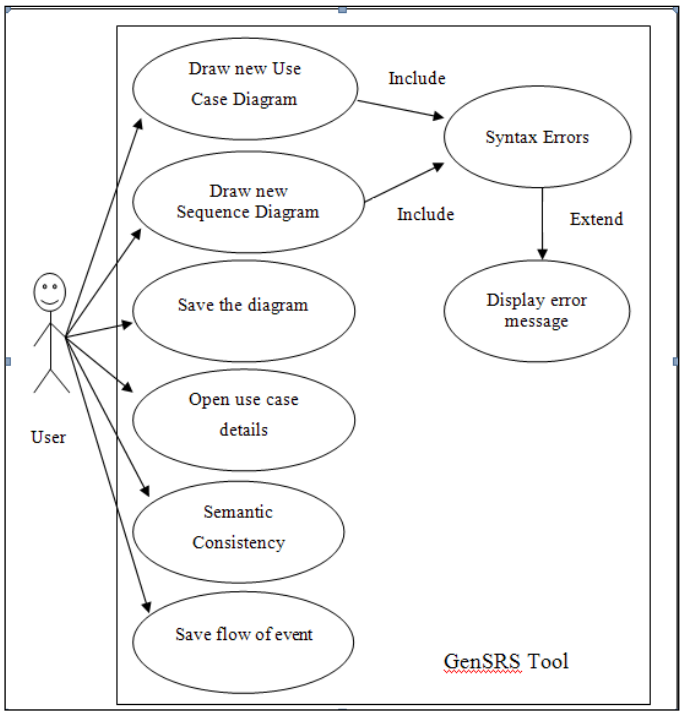

Figure 1. Use case diagram of GenSRS tool

4. Evaluation of GenSRS Tool

This section provides a relatively detailed explanation on how the evaluation the new approach is implemented on consistency check between use case diagram and sequence diagram for check purposes. The case study is a Course registration system IBM 2003 will be introduced. Following this, the implementation of the GenSRS technique is presented, beginning with the use case diagram and sequence diagram of the case studies. Next, the application of the test case generation on the corresponding sequence diagram is discussed.

4.1 Using Syntax rules in Case Study

The tool will verify the syntax errors for all the use case diagram and sequence diagram elements used. In case of any syntax errors in the use case diagram or sequence diagram elements, GenSRS will display an error message to the user. Figure 2, shows the interface of display syntax errors of case study used.

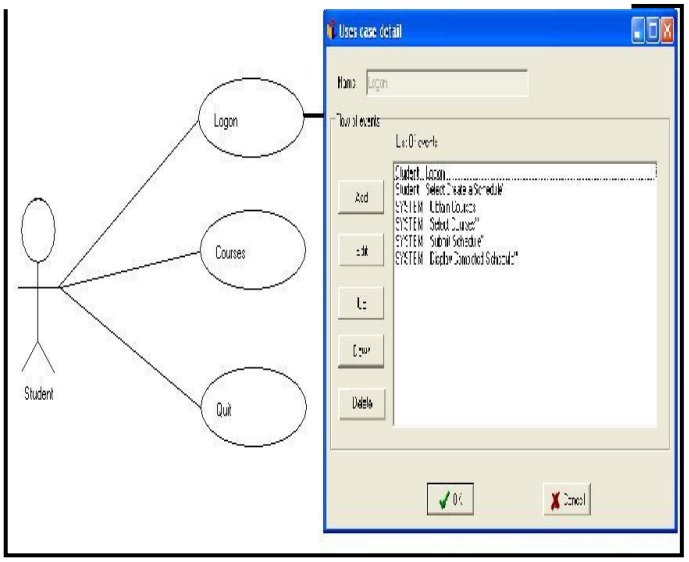

Figure 2: Using syntax errors of use case diagram elements used

Figure 2 shows the interface of display syntax errors of use case diagram elements used because student no relationship with use case (display completed schedule). And rules for use case diagram if no relationship between actor and use case, will display interface (process 6 don't have any connect between use case and actor) in diagram.

4.2 Using semantic consistency in case study

Semantics consistency is used to verify the diagram to capture information from use case diagram. Figure 3 is the interface of displaying semantic consistency of GenSRS tool to applying on case study.

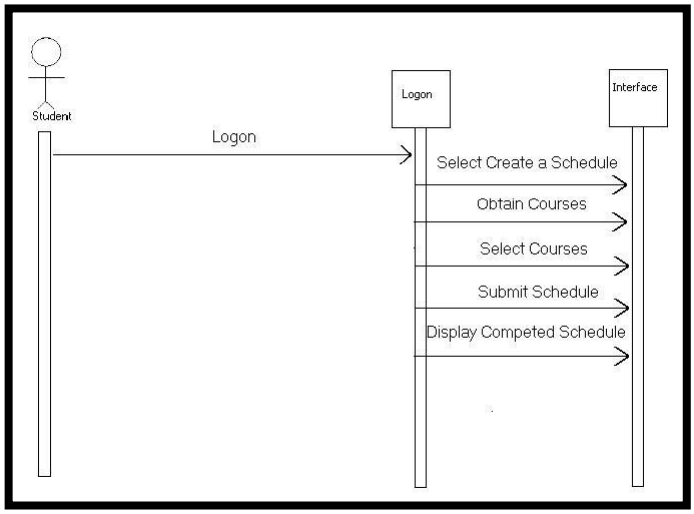

Figure 3: Using semantic consistency on case study

User can use consistency check button on sequence diagram in part semantic consistency based on rules sequence diagram. Semantic consistency will be drowning on the same interface in the sequence diagram by using the specific tools.

4.3 Flow of event in case study

The flow of event in GenSRS to make capture from use case and to save in flow of event. User is able to do six tasks: log on, create a schedule, course information, select courses, submit schedule and display completed schedule. These six tasks represent the use cases of the simple course registration system. From the use case diagram, the flow of events can be constructed. Figure 4, shows the flows of events for the basic flows.

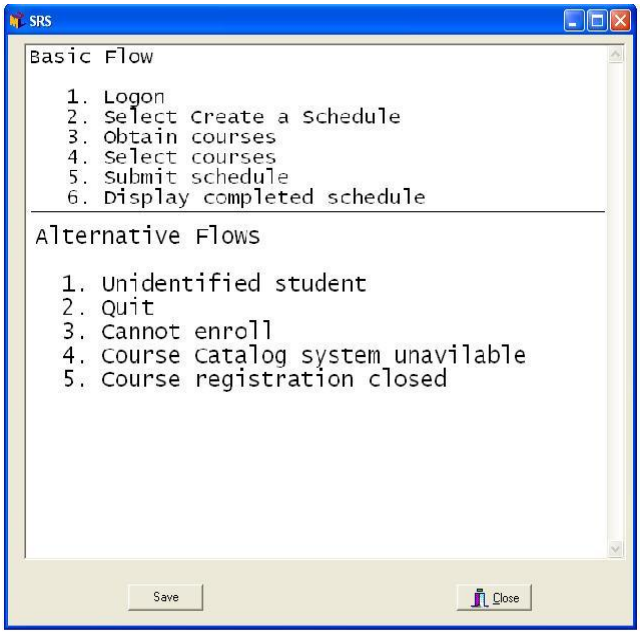

Figure 4: Flow of events for use case Log on case study

Based on the information in the flow of the event, the user can save the flow of event to press button save on TXT file as shown in figure 5.

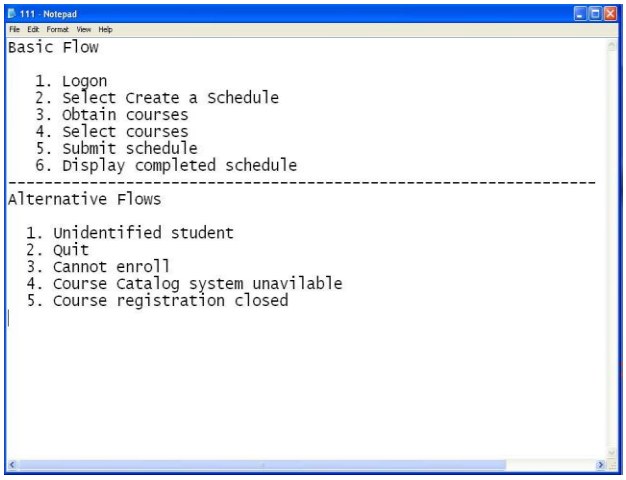

Figure 5: Flow of events saves in TXT file

5. Conclusion and Future Work

GenSRS is a tool that is able to generate Software Requirements Specification (SRS) automatically according to system's requirements. Using use case diagram and sequence diagram to represent a set of syntax errors and semantic consistency of use case diagram and sequence diagram. The rules are then being consistency check between use case diagram and sequence diagram. The first purpose is an editor used to draw the diagrams as well as used to ensure consistency between the diagrams to generate SRS and to save flow of event in TXT file. Future work for this research consists of searching on the possibilities to enhance and improve the functionalities of the tool. Firstly, the tool can be enhanced by enabling the user to draw all UML diagrams. User will be able to draw the use case diagram and sequence diagram by more details. In the same time, it is able to perform syntax error and semantics consistency for all UML diagrams.

The tool can also be enhanced by providing the tools arrangement of the use case diagram and sequence diagram. the saving can be improved in terms of automatically save and update the latest use case diagram and sequence diagram once the user request to generate flow of event.

The tool can be designed to aid user to drawing use case diagram and sequence diagram and in more convenient way. Functions such as copy and paste nodes, group and ungroup nodes, select all nodes and can be applied to the action of drawing UML diagrams. Besides that, allow the user to copy the whole UML diagrams or export the UML diagrams in an image file help the user in reporting.

Acknowledgment

The authors like to thanks UTHM for supporting this research under GIPS.

References

[1] E, eilam,(2005)" Reversing : Secrets of Reverse Engineering", John wiley & Son silne.

[2] Marinos. G, Andreas .S,(2010). Automatic Generation of a Software Requirements Specification (SRS) Document. 10th Internatioanl Confenence on Intelligent Systems Design and Application ISDA. pp.1095-1100, Nov. 29 2010-Dec. 1 2010. Andreou, A S.

[3] Dennis, A., Wixom(2006). Systems Analysis and Design. 3rd ed. Hobken: John Wiley & Sons, Inc.

[4] Davis. Alan, M.(1993), Software Requirements Objects, Functions, and States. New Jersery, Prentice Hall.

[5] Sergin. D, Eric. F, (2007). A Software Tool for Requirements Specification on Using the STORM Environment to Create SRS Documents, ICSOFT International Conference on Software and Data Technologies, vol. 12, pp.319-326. July 22-25, 2007. Barcelona. Spain.

[6] Kimer, T. G, (2000), Applying the SCR method in Software Requirements Specification. International Conference of the Chilean Computer Science Society.

[7] S.Dascalu, E.Fritzinger, K.Cooper, N.Debnath,(2007). A Software Tool for Requirmenets Specification : On Using the STORM Environment to Create SRS Documents, The Second International Conference on Software and Data Technologies, pp. 319-326.

[8] EventStudio,.(2006). EventHelix, EventStudio System Designer 2.5-sequence diagram based system design. Available online as of july 5, 2006 at http://www.eventhelix.xom/EventStudio/.

[9] OSRMT, (2006) Requirements Manangement Tool. News > Available online as of july 1, 2006 at http://www.osrmt.com/

[10] IBM Rational Rose. Available online as of july 12, 2010 at http://www-306.ibm.com/software/rational/.

[11] MagicDraw. Available online as of july 12, 2010 at http://www.magicdraw.com/.

[12] IEEE Std 830-1998, Recommended Practice for software Requirements Specification, IEEE Xplore, 1998.

[13] DOORS, (2007). Telelogic's DOORS> Requirements management traceability solutions. Available online as of Marach 31,2007 at http://www.telelogic.com/products/doorsers/index.cfm.

[14] N. Kassel & B.A Malloy, (2003). An approach to Automate Requirements Elicitation and Specification. Internationa Conf on Software Engineering and Application. Novermber 3-5, 2003, pp. 544-549.