Development of Software Requirement Analysis Tool for NPP Software Fields Based on Software Inspection and Formal Method

Seo Ryong Koo*, Han Seong Son*, Poong Hyun Seong*, Junbeom Yoo**, and Sung Deok Cha**
Korea Advanced Institute of Science and Technology
** Department of Nuclear Engineering*
*** Department of Electrical Engineering & Computer Science, Division of Computer Science*
373-1 Gusong-dong, Yusong-gu, Daejeon, Korea 305-701
**{srkoo, phseong}@mail.kaist.ac.kr, **{jbyoo, sdcha}@salmosa.kaist.ac.kr*

Dae Seong Son and Seong Soo Choi
Atomic Creative Technology Ltd.
106-5 Eoeun-dong, Yuseong-gu, Daejeon, Korea 305-807
1act@hanafos.com

Abstract

This article introduces a computer-aided software requirement analysis tool, Software Inspection Support & Requirement Traceability (SIS-RT), which has inspection, traceability analysis, and formal analysis capabilities. Inspection and requirement traceability analysis are widely believed to be the most effective software verification and validation (V&V) methods. Though formal methods are also considered as an effective V&V harness, they are not easy to be used properly in nuclear fields because of their mathematical nature. These techniques are labor-intensive and thus are required to be partially automated. SIS-RT is designed to partially automate the software inspection process and requirement traceability analysis. Toward easy inspection and effective use of formal method, SIS-RT has three kinds of view; Inspection View, Traceability View, and Structure View. After further development efforts, SIS-RT will turn out to be a unique and promising software requirement analysis tool.

1. Introduction

The use of digital systems is on increase in nuclear industry in recent years. Therefore, the importance of software verification and validation (V&V) is more emphasized in view of the nuclear safety. Inspection is widely believed to be an effective software V&V technique. It can provide a great increase in both productivity and product quality, by reducing development time, through removing more defects than is possible without using inspection, respectively. Inspection applies to the whole lifecycle. By inspecting products as early as possible, major defects will be revealed sooner and will not be propagated through to the final product. However, software inspection is labor-intensive, and it can be difficult to justify the investment in time and money to introduce it. Requirement traceability analysis is to identify requirements that is either missing from, or in addition to, the original requirements. The requirement traceability applied to the software architecture phase can aid in identifying requirements that have not been accounted for in the architecture. Stepwise refinement of the requirements into the architecture produces a natural set of mappings from which to derive the requirement traceability. For large systems, automation is desirable.

Though formal methods, such as Statechart [1], CPN [2], RSML [3], and SCR [4], are also considered as an effective V&V harness, they are not easy to be used properly in nuclear fields because of their mathematical nature. However, formal specification can lessen requirements errors by reducing ambiguity and imprecision and by clarifying instances of inconsistency and incompleteness. In order to promote the application of software inspection and formal method, the authors have been developed a software inspection support tool; SIS-RT. SIS-RT has three views; Inspection View, Traceability View, and Structure View. Inspection View of SIS-RT is designed to partially automate the software inspection process so that the burden of software inspection may be reduced. Requirement traceability analysis, which is considered as one of important activities of software V&V, is supported through Traceability View of SIS-RT. Also, Structure View of SIS-RT supports that the analyzer can easily specify a system using a formal specification method. Therefore, this work suggests an integrated approach with inspection and formal methods in order to support easy inspection and effective use of formal specification method. For the sake of convenience, the authors developed SIS-RT in this work. SIS-RT is expected to be a more effective requirement analysis tool in nuclear fields.

This paper is organized as follows. Section 2 gives the proposed approach for easy inspection and effective use of formal method in this work. Section 3 introduces software inspection and the NuSCR approach. NuSCR is a specification language for describing and verifying a control system for both nuclear engineering and software developer in nuclear fields. In section 4, we present main features of SIS-RT and a brief introduction on three views of SIS-RT. Finally, we conclude our research in section 6.

2. Approach for easy inspection and effective use of formal method

It is very difficult that requirement analyzer understands design documents written in natural language at once and then specifies software requirements from them formally. It is also difficult to assure the quality of the specification. This is because of the difference of domain knowledge between designer and analyzer. Thus it is very important to fill this gap. Design documents written in natural language is mostly of large amount. It needs much time and efforts that analyzer understands and formally specifies the documents.

Using a software requirement analysis support tool which fills the gap between natural language documents phase and formal specification phase, the approach proposed in this research helps a user perform easier inspection and compose formal specification efficiently. Figure 1 shows schematic diagram of the approach proposed in this research.

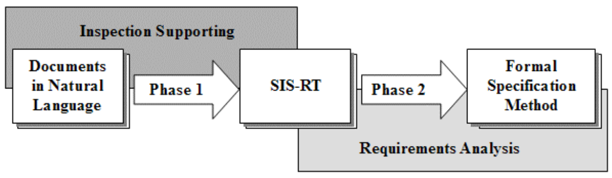


Figure 1. Schematic diagram of the approach

As shown in Figure 1, our approach consists of two phases. Phase 1 is inspection supporting to increase quality of the design documents written in natural language. As mentioned before, SIS-RT supports various V&V activity based on Fagan Inspection [5]. SIS-RT is a PC-based application designed for use by anyone who needs to manage requirements. A desirable attribute of inspections is rigor. Using computers to support the process helps provide this rigor, and improves the repeatability of the inspection process. Repeatability is essential if feedback from the process is to be used to improve it. In phase 1, SIS-RT can support easier inspection for user.

In phase 2, the document analysis feature of SIS-RT enables the effective transition into formal specification. Since it is difficult to generate a formal specification from a natural language document directly, it is necessary to extract useful information from design documents. Structure View of SIS-RT supports the structural analysis of documents. Through the document analysis, we can obtain a refined document for formal specification and this document will be very useful to analyzer. The structure type from the analysis results is affected by formal methods that the analyzer uses for software requirements specification.

In this paper, we propose NuSCR (Software Cost Reduction for nuclear engineering), a specification approach that provides environment to verify the functional requirements of a nuclear control system. The proposed NuSCR provides not only specification approach in specifying requirements but also verification environment. NuSCR is based on the existing AECL approach [6]. It shares the same notation of describing system requirements in Function Overview Diagram (FOD) and in describing each function in tabular notation using the Structural Decision Table (SDT). However, the main purpose of the NuSCR is to reduce the specifying complexity of the AECL approach. The AECL approach describes all requirements specifications based on function nodes in FOD and tables in SDT, which makes timing requirements and history related requirements difficult to specify, whereas the NuSCR uses automata and timed-automata to specify such behaviors that are not easily expressed with the notations of FOD and SDT. In this case, the structure type should be useful to draw FOD and SDT. With an Input-Process-Output structure type, the authors have successfully drawn FOD and SDT using SIS-RT.

3. Introduction to software inspection and NuSCR approach

3.1. Software inspection

Since M.E. Fagan first defined the software inspection process in 1976 [1], there have been many variations of software inspection. We describe here the original method.

An inspection team generally consists of four to six people. Each person has a well-defined role as follows:

Moderator: The moderator is the person in overall charge of the inspection. It is the moderator’s task to invite suitable people to join the inspection team, distribute source materials and to organize and moderate the inspection meeting itself.

Author: The inspection requires the presence of the author of the product under inspection. The author can give invaluable help to the inspectors by answering questions pertaining to the intent of the document.

Reader: During the inspection meeting, it is the reader's job to paraphrase out loud the document under inspection.

Recorder: It is the recorder's duty to note all defects found along with their classification and severity. Although Fagan indicates that this task is accomplished by the moderator, another member of the team is usually chosen, since the workload involved can be quite high, though mainly secretarial. The recorder is often known as the scribe.

Inspector: Any remaining team members are cast as inspectors. Their only duty is to look for defects in the document.

For effective use of software inspection, Fagan describes five stages in the inspection process as follows:

Overview: The entire team is present during the overview. The author describes the general area of work then gives a detailed presentation on the specific document he has produced. This is followed by distribution of the document itself and any necessary related work to all members.

Preparation: Each team member carries out individual preparation, consisting of studying the document to gain an understanding of it. Errors in the document will be found during this stage, but in general not as many as will be found at the next stage. Checklists of common defect types can help the inspectors concentrate on the most beneficial areas of inspection. Each inspector produces a list of comments about the document, indicating defects, omissions and ambiguities.

Inspection: The inspection meeting involves all team members. The reader paraphrases the document, covering all areas. During this process inspectors can stop the reader and raise any issue until a consensus is reached. If an issue is agreed to be a defect, it is classified as missing, wrong or extra. Its severity is also classified (major or minor). At this point the meeting moves on. No attempt is made to find a solution to the defect; this is carried out later. After the meeting, the moderator writes a report detailing the inspection and all defects found. This report is then passed to the author for the next stage.

Rework: During rework, the author carries out modifications to correct all defects found in the document and detailed in the moderator's report.

Follow-Up: After the document has been corrected, the moderator ensures that all required alterations have been made. The moderator then decides whether the document should be re-inspected, either partially or fully.

3.2. NuSCR approach

The Atomic Energy of Canada Limited (AECL) approach specifies a methodology and format for the specification of software requirements for safety critical software used in real-time control and monitoring systems in nuclear generating systems. It is a SCR-style SRS verification method based on Parnas' four variable method. A system reads environment states through monitored variables that are transformed into input variables. The output values of the output variables are calculated and are changed into control variables. The AECL provides two different views of the requirements. A larger view is the FOD and each of the function in it is described by the smaller view of the SDT. The AECL approach specifies all requirements of the nuclear control system in the FOD and SDT notations. This is somewhat complex in cases where timing requirements and history related requirements are considered. This difficulty of specification is modified in the NuSCR approach.

The NuSCR approach is an extended formal verification method of the existing SCR-style AECL approach. The NuSCR specification language was originally designed to simplify the complex specification techniques of certain requirements in the AECL approach. It is an improved method in describing behavior of the history related requirements and timing requirements of the nuclear control system by specifying them in automata and timed-automata respectively. In the existing AECL method, all specifications including history related requirements and timing requirements are specified with only one type of function node in the FOD and with SDT tables. However, the NuSCR uses three different types of nodes in the FOD to specify the properties derived from the requirements. The types consist of nodes that specify history related requirements that are described in automata [7], timing requirements that are described in timed-automata [8], and nodes that specify all other requirements exclusive of the previous two types of functional requirements.

4. SIS-RT

In this section we describe SIS-RT, which is a computer-aided software inspection support tool developed in this work. SIS-RT stands for Software Inspection Support and Requirement Traceability. As mentioned before, we have integrated requirement traceability analysis capability into the software inspection support tool because requirement traceability analysis is considered as one of the items of software inspection. Additionally, formal requirement analysis and inspection meeting support capability are integrated in SIS-RT. That is, SIS-RT composes of a document analysis tool, a traceability analysis tool, a formal analysis tool and an inspection meeting support tool. SIS-RT is designed to support inspection of all software development products. In addition, SIS-RT is a PC-based application designed for use by anyone who needs to manage requirements. Now we describe SIS-RT in view of the features of tool support.

Document Handling: SIS-RT supports document handling very well. It supports cross-referencing from one document to another. As mentioned before, since most inspection documents are produced on computer, it is natural to allow browsing of documents online. Everyone has access to the latest version of each document, and can cross-reference documents using, for example, hypertext. SIS-RT has all these features. SIS-RT can deal with the comments produced by inspectors. They are a major part of the inspection process, as they indicate when an inspector takes issue with a part of the document. SIS-RT allows the comments to be stored on-line, linked to the part of the document to which they refer. They can then be available for all inspectors to study both before and, more importantly, during the inspection meeting.

Individual Preparation: SIS-RT does not have the ability of automated defect detection yet. However, finding them automatically enables inspectors to concentrate on the more difficult defects that cannot be automatically found and that have a greater impact if not found. Thus we are planning to include this capability into SIS-RT. As mentioned before, computer support for software inspection can provide further help during individual preparation in that, by keeping the checklists on-line, the inspector can easily cross-reference between them. On-line checklists can be used by SIS-RT to ensure that each check has been applied to the document. In addition, on-line standards in SIS-RT can assist the inspector in checking a document feature for compliance.

Meeting Support: SIS-RT can help avoid taking many meetings to complete an inspection. By allowing a distributed meeting to be held using web meeting technology, it becomes easier for team members to 'attend' the inspection meeting.

Data Collection: Computer support allows metrics from the inspection to be automatically gathered for analysis. This is a very important aspect. SIS-RT, however, does not have data collection capability. Further development effort for SIS-RT will bring the ability of data collection to it.

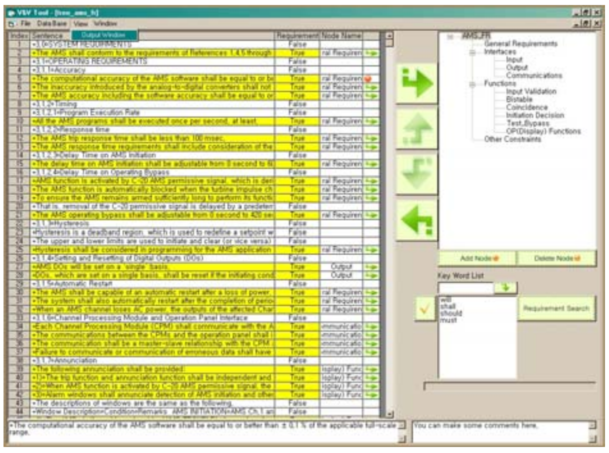
In order to support these features, SIS-RT has three kinds of views; Inspection View, Traceability View, and Structure View.

4.1. Inspection View

The support of document analysis with Inspection View is a main function of SIS-RT. It supports an extraction function that reads a text file and copies paragraph numbers and requirement text to a SIS-RT file. It can read any text data that is convertible to '.txt' format. It also supports manual addition of individual requirements and import from various formats.

Inspection View permits users to associate database items by defining attributes; attributes attached to individual database items provide a powerful means to identify subcategories or database items and manage requirements. Inspection View of SIS-RT supports normal parent/child links to manage requirements. Furthermore, it supports peer links between items in the database and general documents to provide an audit trail showing compliance to quality standards or contractual conditions.

Figure 2 shows a screen shot of the Inspection View of SIS-RT. Inspection View reads source document, identifies requirement, and extracts them for import into the database. Inspection View automatically finds and extracts requirements based on a set of keywords defined by the user. As requirements are found, they are highlighted as shown in Figure 2. The user may also manually select and identify requirements. Inspection View enables us to produce a user-defined report that shows various types of inspection results. Users build up the architecture of the reports that they want to produce on the right-hand side window shown in Figure 2. If a user writes down checklists in the window, SIS-RT can directly support the software inspection with this functional window. Requirements to be found by tool are located in suitable checklist site using various arrow buttons in the window. In this way, each inspector examines requirements and generates the inspection result documents through the supporting of SIS-RT.



Index	Sentence	Requirement	Node Name
1	•3.0•SYSTEM REQUIREMENTS	False	
2	•The AMS shall conform to the requirements of References 1.4.5 through	True	ral Requiren
3	•3.1•OPERATING REQUIREMENTS	False	
4	•3.1.1•Accuracy	False	
5	•The computational accuracy of the AMS software shall be equal to or be	True	ral Requiren
6	•The inaccuracy introduced by the analog-to-digital converters shall not	True	ral Requiren
7	•The AMS accuracy including the software accuracy shall be equal to or	True	ral Requiren
8	•3.1.2•Timing	False	
9	•3.1.2.1•Program Execution Rate	False	
10	•All the AMS programs shall be executed once per second, at least.	True	ral Requiren
11	•3.1.2.2•Response time	False	
12	•The AMS trip response time shall be less than 100 msec.	True	ral Requiren
13	•The AMS response time requirements shall include consideration of the	True	ral Requiren
14	•3.1.2.3•Delay Time on AMS Initiation	False	
15	•The delay time on AMS initiation shall be adjustable from 0 second to 60	True	ral Requiren
16	•3.1.2.4•Delay Time on Operating Bypass	False	
17	•AMS function is activated by C-20 AMS permissive signal, which is der	True	ral Requiren
18	•The AMS function is automatically blocked when the turbine impulse ch	True	ral Requiren
19	•To ensure the AMS remains armed sufficiently long to perform its functi	True	ral Requiren
20	•That is, removal of the C-20 permissive signal is delayed by a predeterm	False	
21	•The AMS operating bypass shall be adjustable from 0 second to 420 se	True	ral Requiren
22	•3.1.3•Hysteresis	False	
23	•Hysteresis is a deadband region, which is used to redefine a setpoint w	False	
24	•The upper and lower limits are used to initiate and clear (or vice versa)	False	
25	•Hysteresis shall be considered in programming for the AMS application	True	ral Requiren
26	•3.1.4•Setting and Resetting of Digital Outputs (DOs)	False	
27	•AMS DOs will be set on a 'single' basis.	True	Output
28	•DOs, which are set on a single basis, shall be reset if the initiating cond	True	Output
29	•3.1.5•Automatic Restart	False	
30	•The AMS shall be capable of an automatic restart after a loss of power.	True	ral Requiren
31	•The system shall also automatically restart after the completion of perio	True	ral Requiren
32	•When an AMS channel loses AC power, the outputs of the affected Char	True	ral Requiren
33	•3.1.6•Channel Processing Module and Operation Panel Interface	False	
34	•Each Channel Processing Module (CPM) shall communicate with the A	True	ommunicatio
35	•The communications between the CPMs and the operation panel shall	True	ommunicatio
36	•The communication shall be a master-slave relationship with the CPM	True	ommunicatio
37	•Failure to communicate or communication of erroneous data shall have	True	ommunicatio
38	•3.1.7•Annunciation	False	
39	•The following annunciation shall be provided:	True	isplay) Func
40	•1)•The trip function and annunciation function shall be independent and	True	isplay) Func
41	•2)•When AMS function is activated by C-20 AMS permissive signal, the	True	isplay) Func
42	•3)•Alarm windows shall annunciate detection of AMS initiation and other	True	isplay) Func
43	•The descriptions of windows are the same as the following.	False	
44	•Window Description•Condition•Remarks AMS INITIATION•AMS Ch.1 an	False	

•The computational accuracy of the AMS software shall be equal to or better than ± 0.1 % of the applicable full-scale range.

You can make some comments here.

Figure 2. Inspection View of SIS-RT

4.2. Traceability View

As mentioned before, SIS-RT supports normal parent/child links and peer links between items in the database and general documents. This is a function related to requirement traceability analysis. Figure 3 shows a screen shot of Traceability View representing the requirement traceability function of SIS-RT. Figure 3 shows that SIS-RT provides mechanisms to easily establish and analyze traceability through the real-time visual notification of change. This capability allows users to pinpoint its impact across the project and assess coverage for verification and validation. Through the Traceability View, we can analyze traceability between source requirements and destination requirements. As shown Figure 3, the column number represents a requirement of source file and the row number represents destination file. The relationships between source and destination are expressed in the matrix window using linked and unlinked chain. That is, the linked chains mean that source requirements are reflected into destination requirements. The unlinked chains represent that source and destination requirements are changed, thus it is necessary to verify the change between source and destination documents. The question marks mean that is difficult to define traceability between requirements. At this time, it is necessary to verify requirements by other analyzer. In order to more easily support traceability analysis, Traceability View has an additional function calculating the similarity between requirements. Through this function, Traceability View can automatically represent the similarity by percentage and then this similarity result is very helpful to user and analyzer. Now, we proposed algorithms to calculate the similarity for both English and Korean documents.

In this way, we can represent the traceability between documents. In the Traceability View of SIS-RT, we can also support the comparing function between requirements. This function helps us to recognize a changed requirement easily.

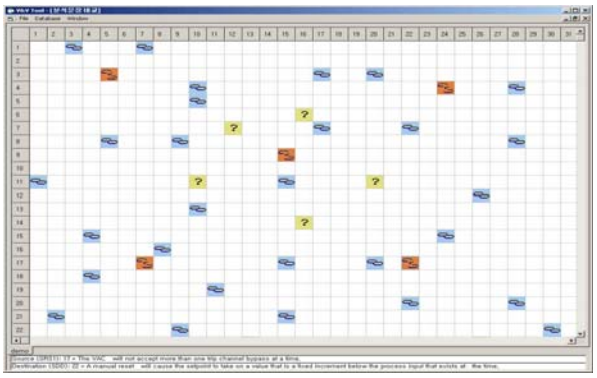
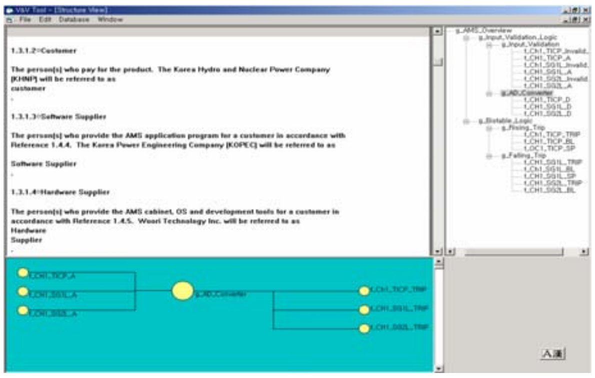


Figure 3. Traceability View of SIS-RT

4.3. Structure View

Structure View of SIS-RT enables the effective transition into NuSCR editor. Figure 4 shows a screen shot of Structure View of SIS-RT. Through the Structure View, we can analyze design documents in view of system's structure and then the analysis results help us generate a formal specification from a natural language document. For the structural analysis of systems, it is the most important to define inputs/outputs and functions. Therefore, we proposed Input-Process-Output structure type in this work. In the Structure View, several tabular forms help users build up Input-Process-Output structure easily and Input-Process-Output structure is represented in right-hand side window as a tree type. After structure analysis, Structure View generates a result file written in XML language and then it is transferred to NuSCR editor. With this file, FOD can be drawn automatically in the NuSCR editor. Figure 5 shows a screen shot of NuSCR editor. The NuSCR Editor is a platform independent tool made with JAVA for formally specifying the SRS of the nuclear control system. It provides environment to draw FOD and SDT and allows automata diagrams to be built from the nodes of the FOD. The Editor also gives a hierarchical view of the SRS described as can be seen on the left side of Figure 5.



1.3.1.2 Customer

The person(s) who pay for the product. The Korea Hydro and Nuclear Power Company (KHNP) will be referred to as customer

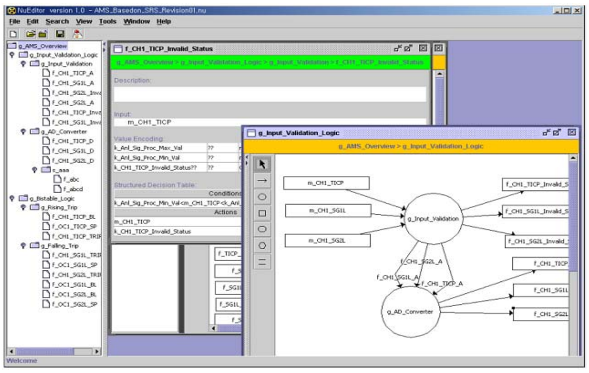
1.3.1.3 Software Supplier

The person(s) who provide the AMS application program for a customer in accordance with Reference 1.4.4. The Korea Power Engineering Company (KOPEC) will be referred to as Software Supplier

1.3.1.4 Hardware Supplier

The person(s) who provide the AMS cabinet, OS and development tools for a customer in accordance with Reference 1.4.5. Woori Technology Inc. will be referred to as Hardware Supplier

Figure 4. Structure View of SIS-RT



NuEditor version 1.0 - AMS_Basedon_SRS_Revision01.nu
File Edit Search View Tools Window Help
f_CH1_TICP_Invalid_Status
g_Input_Validation_Logic
g_AMS_Overview > g_Input_Validation_Logic
m_CH1_TICP
m_CH1_SG1L
m_CH1_SG2L
g_Input_Validation
g_AD_Converter
Welcome

Figure 5. NuSCR editor

5. Conclusions

In this research, we proposed an approach for easy inspection and effective use of formal method and then we developed SIS-RT which is a computer-aided software requirement analysis support tool based on our approach. SIS-RT supports software inspection systematically and has requirement traceability analysis capability. SIS-RT can also support the formal specification using NuSCR editor developed in this work. Through SIS-RT, we can minimize some difficulties caused by the difference on domain knowledge between designer and analyzer. After further development efforts, SIS-RT will turn out to be a unique and promising software requirement analysis tool.

6. References

[1] M.E. Fagan, “Design and Code Inspections to Reduce Errors in Program Development,” *IBM system Journal*, Vol. 15, No. 3, pp. 182-211, 1976.

[2] D. Harel, "Statecharts: A Visual Formalism for Complex Systems," *Science of Computer Programming*, vol. 8, pp.231-274, 1987.

[3] Kurt Jensen, “Coloured Petri Nets (Basic Concepts, Analysis Methods and Practical Use Volume 1), Second Edition”, *Springer-Verlag Berlin Heidelberg*, 1997.

[4] N.G. Leveson, M.P.E. Heimdahl, H. Hildreth, and J.D. Reese, "Requirements Specification for Process-Control Systems," *IEEE Transaction on Software Engineering*, vol.20, no.9, sept. 1994.

[5] C. Heitmeyer and B. Labaw, “Consistency Checking of SCR-style Requirements Specification”, *International Symposium on Requirements Engineering*, March, 1995.

[6] WolsongnNPP 2/3/4, “Software Work Practice Procedure for the Specification of SR for Safety Critical Systems,” *Design Document no. 00-68000-SWP-002, Rev. 0*, Sept. 1991.

[7] J. Hopcroft and J. Ullman, “Introduction to Automata Theory,” *Language and Computation*, Addison-Wesley, 1979.

[8] R. Alur and David L. Dill, “A theory of Timed Automata,” *Theoretical Computer Science* Vol. 126, No. 2, pp. 183-236, April 1994.