*Transactions of the Korean Nuclear Society Spring Meeting*
*Chuncheon, Korea, May 25-26 2006*

# A Study on the Requirements Traceability in the Development of CANDU Core Management Procedure Automation System

Sanghoon Lee, Daeyou Park, Hyungbum Suh*, Sungmin Kim*, Choongsub Yeom
*Institute for Advanced Engineering, Yongin P.O.Box 25, Gyeonggi-Do 449-863, shoon@iae.re.kr*
** Wolsong Nuclear Power Division, Korea Hydro & Nuclear Power*

## 1. Introduction

Requirements traceability is intended to ensure continuous alignment between client's requirements and various outputs of the system development process. *Requirements Traceability* is defined by IEEE Standard Glossary of Software Engineering Terminology as the degree to which a relationship can be established between two or more products of the development process and refers to the ability to describe and follow the life of a requirement, in both a forward and backward directions [1].

The efficient core management and error-free operation are inevitable to increase safety and performance as well as to decrease operational cost. To accomplish these goals many computer softwares were developed and used to relieve and assist lots of complex procedures that should be done by reactor operator effectively and accurately.

As the procedures are getting more complex and the number of programs is increased, an integrated and cooperative system should be introduced. So, KHNP and IAE have been developing a new web-based system which can support effective and accurate reactor operational environment called COMPAS that means CANDU cOre Management Procedure Automation System.

To ensure successful system development, several steps of identifying requirements have been performed and Software Requirements Specification (SRS) document was developed [2]. In this paper we emphasis on how to keep consistency between the requirements and system products by applying requirements traceability methodology.

## 2. Software Requirements of the System

We started to identify software requirements by interviewing members who have responsibility of the reactor management procedures and then wrote SRS document using IEEE 830 template at the beginning of the development.

After making document, we have several meetings and discussion on the requirements to confirm the scope of the system and validate if there were any missing items or not. This kind of process plays very important role in software development process and has close relationship with requirements traceability.

Requirements traceability has to be performed during the whole development phase with configuration management. We adopted bidirectional traceability as follows.

### *2.1 Bidirectional Traceability*

One of the good traceability practices is bidirectional traceability, meaning that the traceability chains can be traced in both the forwards and backwards directions as illustrated in Figure 1 [3].

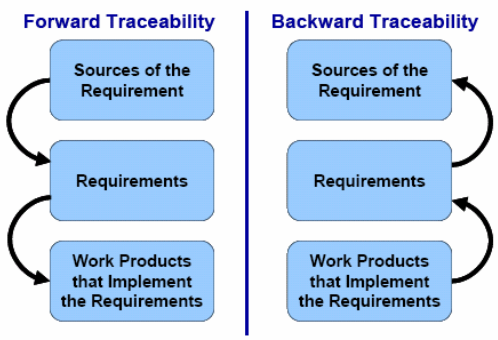


Figure 1. Bidirectional (Forward & Backward) Traceability

Forward traceability looks at tracing the requirements sources to their resulting outputs to ensure the completeness of the requirements specification and each unique requirement forward into the design, implements, code, validation, and so on.

On the contrary, backward traceability looks at tracing each unique product (e.g., design element, object/class, code unit, tests and so on.) reviewing associated requirement. It can verify that the requirements have been kept current with the design, code, and tests. Therefore tracing each requirement backs to its sources.

### *2.2 Benefit of the Requirements Traceability*

As mentioned before, the purpose of *Requirements Traceability* is to manage the requirements of the project product and product components and to identify inconsistencies between those requirements and the project plans and work products [4].

To fulfill the purpose, we adopted bidirectional requirements traceability which has benefits as follows:

- Analyze the impact of change
  - All work products affected by a changed requirement
  - All requirements affected by a change or defect in a work product

- Assess current status of the requirements and the project
  - Identify missing requirements
  - Identify gold plating (products that are not called out in the requirements)

So if we keep traceability of requirements properly, we could make a successful system without difficult and conflict and loss of efforts caused by requirements management.

*2.3 How to Trace Requirements*

The well known and classic way to perform traceability is by constructing a traceability matrix. As illustrated in Table 1, a traceability matrix summarized in matrix form. The traceability from original identified client needs to their associated product requirements and then on to other work product elements.

In order to construct a traceability matrix, each requirements source and each work product element must have a unique identifier that can be used as a reference in the matrix. It can be used as a single repository for documenting both forwards and backwards traceability across all of the work products.

The descriptions on the each fields in Table 1 are as follow;

- User Requirements: initial requirements that initiated current project
- Software Requirements: documentations of user requirements at the beginning
- Design Specification: including class and database schema design with respect to SRS
- Coding component: program source code itself
- Tests: unit test and system test should be performed to verify required function.
- User Manual: the final product which confirm the consistency between system and requirements.

## 3. Conclusion

In this paper, we introduced how we keeping traceability of COMPAS's requirements.

It is very important to ensure that the final product will fulfill the users' requirements. Therefore, we have applied requirements traceability during development to verify current status and outputs including coding, tests, and documentations. We expect that our efforts in keeping traceability of requirements will pay for us by reducing reworks related with wrong or missed requirements.

## REFERENCES

[1] Rosa Candida Pinto, Carla Silva and Jaelson Castro, A Process for Requirement Traceability in Agent Oriented Development

[2] Sanghoon Lee, Daeyou Park, Hyungbum Suh*, Sungmin Kim*, Choongsub Yeom, A Case Study on the Development Process of the Software Requirement Specification of the COMPAS, KNS Conference, Fall, 2005

[3] Linda Westfall, Bidirectional Requirements Traceability

[4] CMMI for Systems Engineering/Software Engineering, Version 1.02 (CMMI-SW/SW, V1.02); CMMI Staged Representation, CMU/SEI-2000-TR-018, ESC-TR-2000-018; Continous Representation, CMU/SEI-2000-TR-019, ESC-TR-2000-019; Product Development Team; Software Engineering Institute; November 2000.

Table 1. Example of a Traceability Matrix

| User Requirements | Software Requirements | Design Specification | Coding Component | Test Case # | User Manual |
|---|---|---|---|---|---|
| RFSP execution | SRS 3.1.1 RFSP Run | HLD* 3.1.3 LLD** 49 | RFSP_RUN.cs RFSP.aspx | TS 3.1 | Section 3.1 |
| | SRS 3.1.1.2 RFSP Pre-simulation | HLD 3.1.33 LLD 51 | RFSP_PRE.cs RFSP_PRE.aspx | TS 3.1.1 | Section 3.1.1 |
| Gateway Data Extraction & Import | SRS 3.4.3 Gateway Data Mgt. | HLD 3.14.1.2 LLD 108 | Gateway.aspx Gateway.aspx.cs | TS 3.14 | Section 3.14 |

*: High Level Design, **: Low Level Design