Software lifecycle support and management system for safety-related signalling system

Lei CHEN

Certified Safety Engineer of P.R.China

Beijing National Railway Research & Design Institute of Signal & Communication Co., Ltd.

SUMMARY

This paper presents a software lifecycle support and management system for safety-related signalling system, which combines mature European safety assurance concepts and best practices and is fully compliant with the new EN50128. Based on human factor engineering and task analysis techniques, the system is a tentative effort to solve the contradiction of limited time of a project and high RAMS performance requirements to some extent. The evaluation of the tool shows that it can improve efficiency and reduce deliver time of the project without any compromise on software function, performance and safety.

INTRODUCTION

Safety is such a major issue in railway that passengers, customers, operators, and contractors are all concerned about. With massive constructions of high-speed railway in China, how to develop or configure a safety-related signalling system and control the residual risk of the system under an acceptable level within a demanding schedule challenges all the signalling contractors.

Being the most important signalling system supplier and system integrator in China, Beijing National Railway Design and Research Institute of Signal and Communication Co., Ltd. (hereafter: CRSCD) takes safety as its primary goal. An organization-level Safety Assurance System (SAS) fitting for all safety-related R&D projects and system integration projects has been released and implemented since 2010. It has greatly improved the safety management of CRSCD. The traditional way of project management and safety assurance always costs time and resource, which prolongs the development period and reduces the competitiveness of a product of a project in this fast developing environment.

This paper presents a software lifecycle support and management system for safety-related signalling system (SwLSMS), which is a tentative effort of CRSCD to solve the contradiction of limited time of a project and high RAMS performance requirements to some extent. The following sections are focused on major challenges we met, task analysis on software development with human factor engineering method, and a detailed introduction on the SwSLMS.

MAJOR CHALLENGES

To develop safety-related signalling system software, there is more additional work to do than ordinary software development. It seems that the safety performance of signalling system contradicts with software development efficiency.

The traditional way of project management and safety assurance always need different roles to act together to make reliable decisions. The project must take a lot of time and resources to coordinate activities of different people. It prolongs the development period and reduces the competitiveness.

Furthermore, requirement driven development and testing approach is adopted throughout software lifecycles. But tools used in different software lifecycles form many information islands. They cost much

human work which often induces mistakes to link these information together and maintain these traceable links.

The coding assist tools are widely used for programing. For safety-related signalling software, there are many special coding standards and coding style requirements. Without instant reminding and development assistance tools for these requirements, some mistakes often won't be found until verification or testing process.

To find bugs of the system as far as possible, a large amount of testing cases in different levels must be designed and performed. One round of full test of a system always needs a lot of testers and takes time. Testing always is the most time and recourse consuming phase of software development or configuration especially when software often changes due to design changes or debugging.

Although knowledge base for software development was set up in CRSCD, these information are stored and shared by individual documents, among which there are no interrelationship. It is hard to find an effective and credible way to make use of existing components most of which are proven in use.

TASK ANALYSIS APPROACH

Although the problems we met listed in the previous chapter have been lasted for many years, the CRSCD is devoted to finding a solution.

Task analysis is the analysis of how a task is accomplished, including a detailed description of both manual and mental activities, task and element durations, task frequency, task allocation, task complexity, environmental conditions, necessary clothing and equipment, and any other unique factors involved in or required for one or more people to perform a given task[1]. A task analysis is a method for developing an understanding of the structure of a task that a person is required to do, which is always used before risk analysis to break down a complex task into simple actions.

According to the new version of EN50128, the software lifecycle has been divided into 7 stages in CRSCD. 29 high level tasks are identified in the overall software lifecycle, which include software assurance, development, data configuration, testing, verification, and validation activities. Each task that the user will perform needs to be systematically broken down and understood. By investigating stakeholders of the software lifecycle, tasks are broken down into task areas, and finally into simple actions with functions. There are more than 800 actions in the 7 stages.

Based on human factor engineering research, how human interact with the computer and the environment to complete each tasks are detailed studied. Many attributes are added to indicate properties of actions. The key attributes are listed below.

- Pre-actions: identify all actions which are the precondition of the action being analysed.
- Needed information: identify all information needed to do the action.
- Effected actions: identify all actions may be effected by the action.
- Who's responsible: identify the role that's responsible for the action.
- Done automatically by computer: the action is done by computer or human.
- Non-conformities: statistical study on non-conformities related to the action in the recent 3 years.
- Can be done by computer: investigate the possibility to automate the action by computer.

By interviewing with roles involved, these attributes are decided according to the reality of CRSCD and the computer technology.

The following figure shows a part of the hierarchical task analysis (HTA) result of document review application task area.

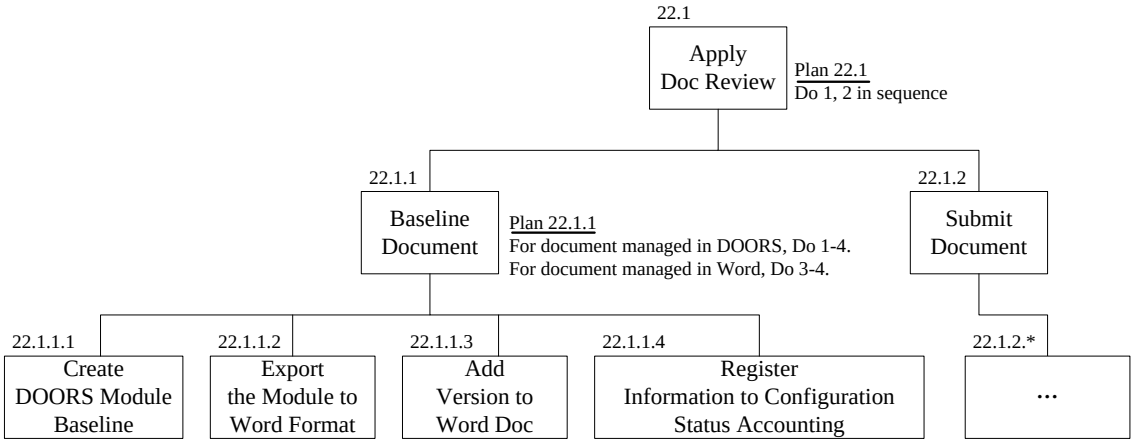

Figure 1: HTA for document review application

In CRSCD, most requirement documents, testing documents and some design documents of newly developed signalling systems in recent years are managed in IBM DOORS in order to maintain the traceability. Review process is built on other software platform, and the document needs to be word format for the convenience of reviewers. Taking account of action numbered 22.1.1.4 for instance, the HTA result is shown below:

Action No.	22.1.1.4
Pre-actions	22.1.1.1, 22.1.1.2
Needed information	DOORS Module URL and baseline number Word document version number, SVN URL and Revision
Effected actions	22.1.2.*
Who's responsible	The writer of the document
Done automatically by computer?	No, human
Non-conformities	27 non-conformities for word and DOORS information mismatch or leave out in CM audit in R&D centre
Can be done by computer	Yes

After analysis, all actions are divided into two groups by attribute "Can be done by computer". One group which is routine work or calculation work is fit for computer. Only actions which need experiences or innovations are left for the project members.

THE TOOL

The original intention of CRSCD to develop the tool is to support developers to increase efficiency of design and coding activities, and introduce UML to standardize the design descriptions. After the organization-level safety assurance system has been released and implemented since 2010, the development of new signalling system costs more time and more resources, which may influence the competitiveness of new products of CRSCD. So CRSCD expands the tool project to build a software lifecycle support and management system for safety-related signalling system. The SwSLMS is an attempt to solve the challenges I have mentioned above, in which many innovations are made.

All newly added actions done by computers from the results of HTA have been input to the SwSLMS as important requirement. To automate all these actions by computers will greatly shorten the duration of tasks related, and reduce human errors in these tasks.

The architecture of SwSLMS

The SwSLMS is a tool to support all activities in the whole software lifecycle, including development, management and assurance activities and processes. It provides requirement development, architecture design, component design, coding functions for developers, integration, testing, review and verification functions for integrators, testers and verifiers. It enables different people to act together in a cooperative manner to increase efficiency of producing new software. Quality and safety management processes related with software development are embedded in the tool. All activities and processes are recorded in the SwSLMS, which make it easy for conducting quality and safety assurance. All data in different databases are linked together to make maximum use of project information to assist management decisions.

The SwSLMS has a 3 layer structure in functional level, which are presentation layer, core business logic layer and key support service layer. An illustration of functional architecture of the tool is shown in figure 2.

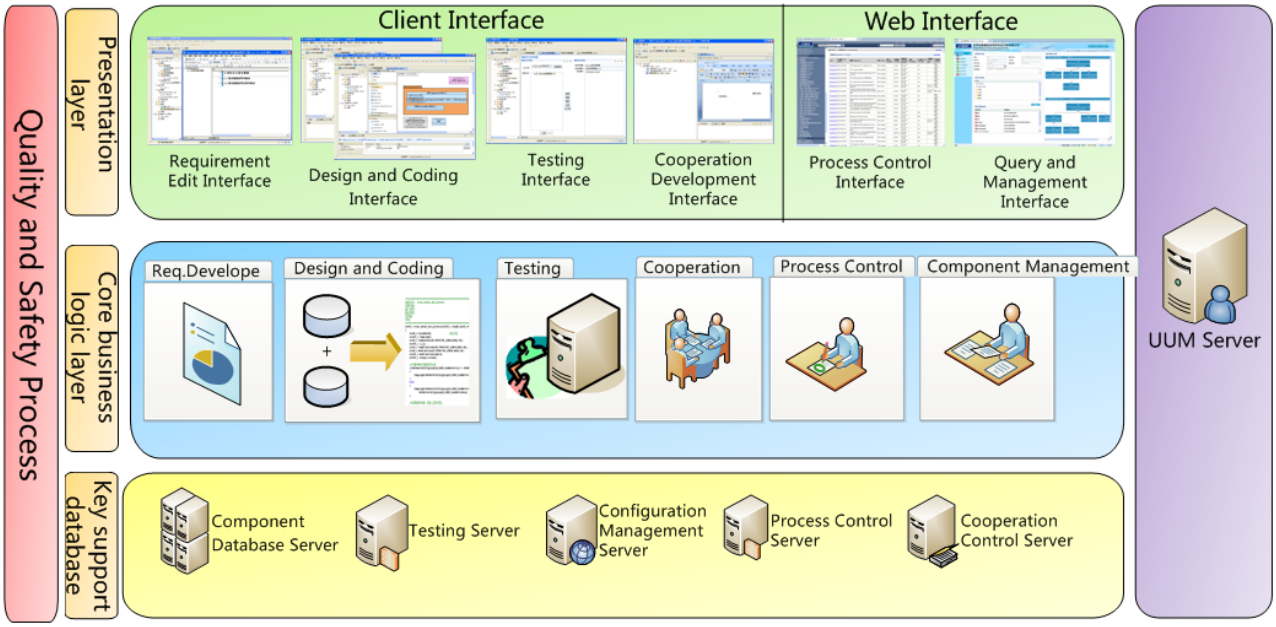

Figure 2: Functional architecture of SwSLMS

As is shown in the figure, a unique user management and authorization service and quality and safety management processes run through the 3 layer structure.

The layer interacting with users is presentation layer. For users of SwSLMS, the tool provides client-server approach and browser-server approach for different roles and purposes on the study results of human factor engineering. The client interface is designed for the project implementation team, which gives multiple functions and friendly operational experience to facilitate users to concentrate on their own work. The client part of SwSLMS is developed on the basis of Eclipse framework, so the new tool keeps the original style of the Eclipse interface, which is quite familiar for the users. For the management team and quality and safety assurance team, an easy-to-access and rich-expressive interface is needed. The web interface is the best solution. The layout and information provided are redesigned and optimized in the SwSLMS, but the style of the web interface remains the same as the existing office automatic system and the IBM Change system for the convenience of the users.

The core business logic layer provides all necessary functions to support software lifecycle activities and management decisions. The main functions include requirement development, design and coding, testing and integration, cooperation work and configuration management supporting, process control and component management. Functions need information exchange and data access, so they must synthesize data from different database and make proper use of data.

All data are stored in the key support database layer. New services and databases are developed to support functions in the upper layer. Component database server provides query and knowledge base service.

Testing server is the kernel of test function. In order to make use of all historical data of CRSCD, main databases have been kept compatible with original ones. That means the SwSLMS must handle data from different type of databases, such as Synergy database for IBM Change process, subversion database for file management and DOORS database. To synchronize these databases, configuration management server manages a batch of datasheets to link configuration management information from independent databases together. This server can complete configuration management based on tasks, give configuration status report automatically, and support cooperation from different persons and different roles. The process control server integrates the change requests managed by IBM Change, and processes in the existing office automatic system.

Key innovations

To handle the challenges we met, there are many key innovations in the development of the SwSLMS.

1. Quality gate control

A quality gate is a special milestone in a software project. These are located before a phase that is strongly dependent on the outcome of a previous phase. Each quality gate includes a formal check of documents and their status relevant to the previous phase. To automate this control method, a unique user management and authorization server takes control of all software related using Lightweight Directory Access Protocol (LDAP) and overall access control. The SwSLMS integrates access and role control, review and authorization management to ensure that only after all requirements for passing the quality gate are met, can a project go on to the next stage.

2. Information fusion

The following figure shows how the SwSLMS deals with existing system.

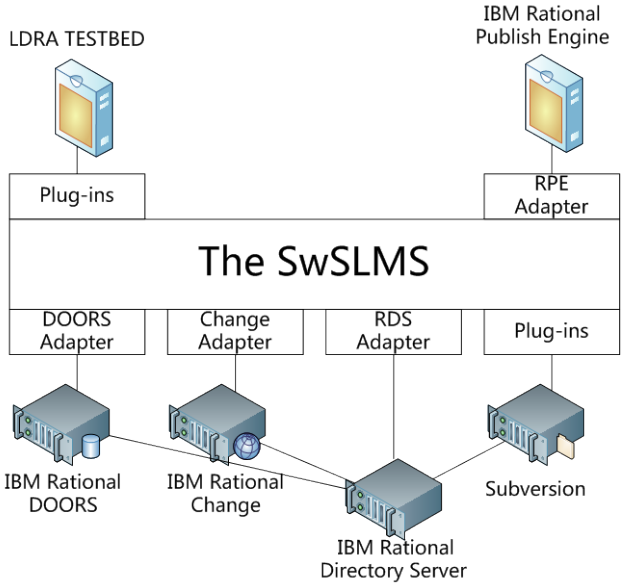

Figure 3: Relationship between SwSLMS and other systems

Information flow during software lifecycle stages is studied to identify any linking requirements of data from different databases. By making maximum use of existing tools, specific tool adaptors and plug-ins are developed and information fusion techniques are adopted. The SwSLMS can make data from different sources easily synchronized according to the need of management. For example, the SwSLMS can manage the traceability of requirement specifications, designs, coding tasks, test cases, test scripts, test records bidirectional and produce traceability table automatically.

A gap analysis was conducted to find any management gap of existing tools against the requirements of EN50128. New services and tools, such as configuration management server, are developed to fill the gap

among IBM DOORS, Change and subversion tools. The SwSLMS can support all software lifecycles and main techniques defined in EN50128.

3. Development assistance for safety-related signalling software

According to the most needed assistance functions from project survey, collaboration development tool, code completion tool, instant code rules check-up tool, cross compile tool and debugger are presented and integrated, which make it easy to work and improve the quality of stage deliveries.

The instant code rules check-up tool has realized 60% of the coding standards and coding style requirements for general safety-related signalling software. It's an effective way to find non-conformities against coding standards before static analysis made by testers.

4. Automatic testing

The static analysis, dynamic analysis and testing on white box level are already automated by using LDRA Testbed tool. So we are mainly concerned about how to make functional testing and performance testing auto-executed. Considering the characteristic of signalling system, a unified test script driving mechanism for all software tests is designed (patent), and testing tools are developed. Figure 4 illustrates the logical structure of automatic testing mechanism.

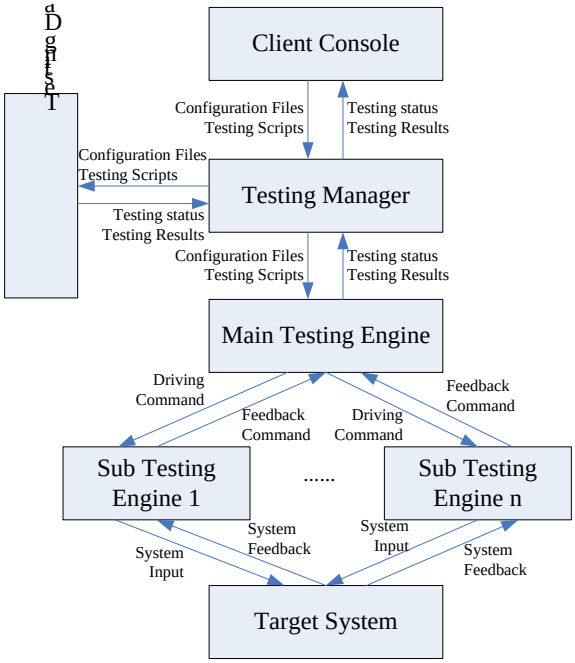

Figure 4: Logical structure of automatic testing mechanism

The client console is a graphical user interface for testing. Testers can use the console at their own SwSLMS client to configure testing environments, edit testing scripts, control testing process, observe testing status and results, and export testing documents. All testing related data are stored in the testing database. Testing manager is the control centre of testing, which can manage up to 64 main testing engines. One main testing engine with a serious of sub testing engines forms a testing driving system for a target system. Testing manager gets testing scripts and configuration files from database or interface according to the testing schedule, and sets these information to the specific main testing engine. During testing, testing status and results are feedback to testing manager, and then finally stored in database or displayed in user interface. Taking Train Control Centre (TCC) testing as an example, TCC related system, such as CBI, LEU, track circuit, TSR server, and neighbouring TCC should be simulated by sub testing engines one by one. To target system like TCC, sub testing engines act like real systems communicating with TCC. The sub testing engine receives commands from main testing engine and converts them to the format as real system. When the TCC gives response to a system, such as LEU, the sub testing engine simulates the LEU to convert the feedback to a command, and sends it to main testing engine for testing result decision.

Automatic testing tool releases the working load of testing executing, which will save a lot of human resources in regression testing and facilitate debugging and regression testing.

5. Component based knowledge base

A new concept of COMPONENT is defined (patent), which encapsulates a functional modular part of system, including software requirements, interfaces, designs, codes, test cases, test scripts, test records and relevant documents. The component database server is developed, and a component share policy is designed, which forms a knowledge base. When a component is confirmed by the sharing process, all authorized projects can easily make use of it in a new development. One component in the base can be reused in different stages and levels. The structure of a component can make different project use one component in its own way, which greatly facilitates the reusing of proved knowledge.

Trial use

The SwLSMS tool is on trial in some projects of my company, such as zone control certre(ZC) software of CBTC system. These projects are in progress, so there are no final statistic data.

Project members and project managers all give the SwLSMS positive comments. The ZC project is at the software test stage. Now the number of functional test cases is 694. With the help of the SwLSMS, one round of full software functional test costs only 8 machine hours. The project uses daily build and testing method to find any deviation as early as possible, which greatly increase the quality of milestone deliveries and shorten the delivery time.

CONCLUSION

The software lifecycle support and management system for safety-related signalling system combines mature European safety assurance concepts and best practices and is fully compliant with the new EN50128. Developing the SwLSMS is a tentative effort of CRSCD to solve the contradiction of limited time of a project and high RAMS performance requirements to some extent. The tool is on the bases of human factor study and task analysis of all activities of the software lifecycle. The evaluation of the tool shows that it can improve efficiency and reduce deliver time of the project without any compromise on software function, performance and safety.

References

[1] Kirwan, B. and Ainsworth, L. (Eds.). A guide to task analysis[D]. Taylor and Francis, 1992