An experience of using RACME tool for the V&V process in aerospace domain

Lorenzo Vinerbi, Fabio Duchi
ResilTech, Italy
{lorenzo.vinerbi, fabio.duchi}@resiltech.com

Fátima Mattiello-Francisco
National Institute for Space Research - INPE, Brazil
fatima@dss.inpe.br

Andrea Bondavalli
University of Firenze
UNIFI, Italy
bondavalli@unifi.it

Abstract

Together with the increasing complexity of Critical Systems, an increased interest in safety standards has been observed in the latest years. In order to support experts on Verification & Validation (V&V) tasks, automated tools are being developed. RACME is a tool designed to import, interpret and correlate the different outcomes of the V&V activities. The conception and the initial design of the tool are presented in [1]. In this paper we report the tool evolution during last year with improvements to address a completely different application domain. Indeed we focused on a project with very different characteristics from the ones met in the past that has allowed us to better understand the needs related to the flexibility and the extensibility of the tool. The work was useful to re-consider some of the subsequent development steps of RACME, mainly concerning: i) definition of the workflow characterizing a project; ii) improvements on the input/output plugins (with a particular attention to the scripts development); iii) improvements on test management (possibly with the support for system modeling tools).

Keywords
Verification & Validation, documentation, certification, standards, methodology.

1. Introduction

Several different V&V processes exist and are currently applied, depending on the application field considered and the applicable standards. Each standard requires and suggests a specific process, specific skills, technology and supporting tools. Despite the existence of this large number of V&V activities, the number of frameworks or platforms that support the overall V&V process is very limited. Indeed, we have identified only few solutions which are able to: i) guide the overall V&V process in agreement to standards; ii) interconnect the results of various activities and iii) organize the V&V outputs and provide certification evidence.

RACME (Resiltech Assessment and Certification MEthodology) [1] is a framework that copes with the previous issues and proposes a methodology and related instruments to support V&V processes and certification activities.

The main progresses intended with the RACME methodology are summarized by the following points:

i) identification, monitoring and control of all safety and certification related activities and information within the system life-cycle;

ii) improvement of the traceability, including artifacts from all the system development life-cycle; and

iii) leveling of the presentation, data and control layers with a deep enhancement in terms of usability and, consequently, correctness of the information treated.

Before describing the accomplished work (Section 2) and the obtained results (Section 3) we are going to better clarify:

i) the past experiences on RACME (Section 1.1), in order to understand our background with this tool;

ii) what is the QSEE project (Section 1.2) and the reasons which let us consider it a good case study for our tool.

1.1. Past Experiences on RACME

The conception of RACME began in 2009, whereas the implementation of the core began in 2010. The July 2012 release is currently being evaluated on two industrial projects (related with railway context).

In [1] a case study related to the European ALARP project on an Automatic Track Warning System was described. Despite the early stage of development, RACME could provide to ALARP a relevant contribution to the V&V process in the following three
areas: management of the documentation, management of the input information, management of the hazard log. In the following months the development team focused on enriching a working prototype with the realization of the test suites designed to validate the correctness of such implementation.

1.2. The QSEE Project

The QSEE (Qualidade do Software Embarcado em aplicações Espaciais [2]) is a project executed at the Brazilian National Institute for Space Research (INPE) from 2005 to 2008. The target of QSEE was the development of particular satellite embedded software, named SwPDC, by a Brazilian industry supplier from Information Technology sector. Despite CMMI level III qualified, the supplier was not familiar with the development of space embedded software and the ECSS (European Cooperation on Space Standardization) standard followed by INPE’s space projects [5].

QSEE is a significant case study for RACME, because: i) it is a small project focused on a reduced number of V&V activities to accomplish; ii) RACME has never been used before on aerospace projects, so it represents a challenge where RACME can be evaluated; iii) QSEE focuses on the testing phase [3], whereas railway processes, and thus earlier releases of RACME, focused on hazard analysis and requirements traceability [4].

2. Using RACME In QSEE

In the following sections we report the main results of the work done, related to: i) the specification of the workflow adopted within the QSEE project; ii) the characteristics required to handle the QSEE model-based testing documentation.

2.1 Workflow in QSEE

A V&V process within RACME is built upon the concept of workflow, which is a sequence of activities to be carried on. Each task can be characterized by its own properties and may be linked with other activities to define a graph that describes the dependencies between activities (in terms of priorities and I/O).

RACME has been developed to be usable with different standards. In RACME the reference workflow for a V&V process is specified in a configuration file that allows the tool set-up and the set-up of the plugins involved in the V&V process. The RACME support to ECSS project has required the writing of a specific workflow configuration file compliant with ECSS standards.

Figure 1 and Figure 2 show a part of the customization for the QSEE project of the general workflow proposed in [1].

The boxes filled in blue highlight the input documents for RACME customization for space projects in order to follow ECSS standards (each project that have to be compliant to ECSS must have these boxes on the reference workflow). The other boxes partially blue highlighted concern to QSEE specific documents and customized activities.

Figure 1: QSEE Workflow in RACME (part 1).

Figure 2 shows the part of the workflow related with the test management. The links between “Select V&V activities”, “Model-based Test supporting methods and tools” and “Test execution supporting Tool” emphasize that RACME has to be capable to receive information from the tool used to produce test specification and the tool used to execute the tests.
Here we noticed the first issue in RACME customization to QSEE project. Indeed, people in charge to define QSEE workflow had difficulties on the file editing task because it: i) requires knowledge of the XML dialect used in RACME to describe the different objects and their properties; ii) is an activity very slow to accomplish; iii) is easy to make mistakes. These remarks led us to investigate the adoption of a graphical language for the workflow specification.

2.2 Development of the Plugin For Test Management

Although RACME provides a library of plugins that covers some aspects of a V&V process, an advanced support for test management still lacks.

The testing activity in the QSEE project has characteristics very different from those encountered in ALARP [1], mainly because of the model-based approach adopted in QSEE project. These considerations have led us to improve the RACME module for supporting test management in order to be useful in QSEE. This has been relatively easy since RACME is characterized by a pluggable architecture.

The testing module has been designed in order to:

i) import automatically test specifications and their execution descriptions;
ii) import automatically tests reports;
iii) manage related information such as test execution result, executor (V&V operator that has executed the test), date and time of execution;
iv) make an automatic versioning of these information in order to provide the history of tests execution;
v) produce reports about the actual state of the testing activity.

In order to add such module, we spent 2 weeks of work, that can be considered as a good result, which confirms that the architectural choices were good.

A module has been developed in order to transforms HTML, which is the original format of the QSEE document, to XML, which is the format natively handled by RACME. Indeed, an XML dialect has been defined in order to: i) represent the structure of the original document; ii) contain the textual data of the original document.

Below, an example for requirements importing is given:

```xml
<word-document/table [cell/paragraph/@style = "Req_ID"]
```

Information contained in the XML document are recognised by their structure and their format as well (XPath is the technology behind data navigation). Information to be imported in RACME was mainly related to tests (test specification as well as test report).

In QSEE these information are produced automatically by the test case generation tool (test specification) or by the test execution tool (test report). The patterns for retrieving test id, body, executor, report and other information have been defined easily and it have not raised any issue. In ALARP, on the contrary, the different partners had used different formalisms to define test specifications and reports. This fact produced very heterogeneous documents that required lot of time in development of importing scripts.

2.3 Scripts Development

Scripting-engines offer an easy way to extend an application (since script development is less expensive in time than a plugin development). On the other hand, they always slow down the application. In order to balance the performance and to speed up the customization process, RACME uses an hybrid approach. The extensions of RACME functionalities that require high performance and that are potentially reused in many other projects are written as plugin; the extensions that fit to just a single project, like the ones used for transforming specific raw data into information (which are strictly related to the structure of documents), are written in LUA [6].

The time spent in customizing RACME for its usage in a project is a key point for RACME tool. The scripts development represents an important time consuming activity. In the context of the QSEE project, the documents have a completely new structure. This fact allowed us to evaluate the time needed by a V&V expert, with a basic knowledge of computer programming, to write a LUA script interacting with RACME. A little team of two people has been set up aiming to provide a script able to convert the raw data to information suitable for RACME.

LUA, like many other scripting languages, has a learning curve that grows quickly. A couple of days were necessary for both of our V&V experts to reach an acceptable knowledge of LUA. Besides that, 5 days were spent, in the worst case, to achieve the goal. The difficulties identified are related to the characteristic of RACME as a tool useable in different contexts. Many objects accessible by LUA are depended to the project for which the script is written.

These thoughts have led us to the development of an IDE designed specifically for speeding up the writing of LUA scripts; providing a debugging environment with real time objects hints, breakpoints, variables and stack trace inspection.

In order to evaluate the improvement given by the new IDE, we have spotted a new team of two people, with the same goal of the others.

The experience shown that the time spent for developing the script pulled down to just 2 days,
surprisingly even the time spent for learning the language has decreased to 1 day.

3. Conclusions

In this practical experience report we have highlighted the main topics raised during the RACME customization for the QSEE project.

Our analysis of the QSEE project focused on three key fields that a tool such as RACME needs to address:

i) customization: it has to be flexible and powerful in order to drive the tool through the rules defined by the V&V standards;
ii) integration: it has to adapt itself in order to allow the different V&V activities to share results and data;
iii) extensibility: general activities can be replaced with ones specifically designed.

3.1 Customization

Although the workflow for QSEE has been clearly defined, some difficulties have been experienced during its definition: in particular we noticed that a big support to the V&V expert from a developer is needed, which lead to spend a lot of time in this activity: indeed we used 3 days to completely define workflow, with minor increase with respect to ALARP case study (which required 3.5 days). This difficulty is mainly related with the lack of a well-defined and largely adopted language for workflow definition. We think that the adoption of such a language could really improve usability of this part of the tool.

3.2 Integration

The RACME customization for QSEE project has highlighted how much projects belonging to different contexts can differ from each other.

The integration activity has been solved by using the LUA scripting engine. Scripts have been developed in order to build a bridge between activities and RACME. This challenge has been addressed by providing RACME with an IDE specifically designed. This solution simplifies the code writing and encourages the code reusability. If we compare time needed to develop a script we used in ALARP (5 days), with the time spent for a similar script developed for QSEE project (2 days), we notice a good improvement.

3.3 Extensibility

We also have extended RACME with the introduction of an improved plugin for tests management. The extendibility degree offered by the plugin-architecture, upon which RACME has been built, has proved to fit the needs of the ECSS project. We also have experienced that the effort involved for the development of the plugin is not as relevant as we thought (about 2 weeks). Moreover, the plugin development is expected to be an activity more and more rare with the evolution of the tool, because a library made up of the already-developed plugins will be kept available.

A survey has been conducted among the users of RACME. Even though they could not try all the feature of the tool, because this case study was mainly focused on test management, they have reported the tool has improved their work, in particular for what concerns data input, versioning and representation. Even if it is difficult to exactly estimate such improvement (the only way should be the usage of two team working on the same project), we could say that about 3 days have been saved.

Summarizing, if we evaluate this experience only under the point of view of the time used to complete the tasks, we haven’t had positive feedback; however have to be considered that: i) RACME is a very young tool; ii) a lot of work done here will be reused in future projects; iii) we made experience and we learned a lot on the actual lacks of the tool and on the directions of work to be taken.

4. References