Capturing Architectural Artefacts in a Configuration Management tool

Vittorio Torroni
Serco Italia
Frascati, Italy

***Abstract*— too many architectural artefacts are lost in the transition from the Design to the Operations phase. This paper shows how to extract key architectural information from a System Model or set of documents and translate them into artefacts that can be added to a Configuration Management tool to facilitate change impact analysis.**

Keywords—Configuration Management; Transition Stage; System Modelling; Change Impact Analysis; Service Management

I. Introduction

The Systems Engineering (SE) good practices described in the INCOSE SE Handbook [1] focus on the activities performed in the Concept and Development stages, in order to avoid discovering costly mistakes later in the Production and Utilization/Support stages.

Unfortunately, in practical applications, this emphasis on requirements and design often translates in using the SE approach and methods in the requirements, architecture and design definition processes, and abandoning them in the Operations and Maintenance processes. A typical symptom of this approach is that important System Model information (regardless of whether they were defined in a System Modelling tool or in architectural documents) are not handed over to the Operations and Maintenance (O&M) teams. For example, if the Design team used a modelling tool, the O&M team often does not perceive the value of using it to support O&M processes.

This approach makes it difficult to perform Change Impact Analysis as part of the O&M processes. What O&M teams do always use is a Configuration Management (CM) tool, which may be integrated with other tools, such as Anomaly Management, Documentation Management, etc.

This paper presents the results of a case study, performed in the framework of the Operations and Maintenance of the CryoSat-2 Payload Data Ground Segment (PDGS), managed by the European Space Agency (ESA), directorate of Earth Observation Programmes, Ground Segment Infrastructure and Operations Management Division.

The paper is organized as follows:

- Section II provides a description of the CryoSat-2 PDGS context, the architectural information handed over by the Design team, and the main features of the CM and Service Management tool used to support the the O&M processes;
- Section III describes the process that was adopted to translate the need to perform automated and robust Change Impact Analysis into specific updates applied to the CM tool;
- Section IV illustrates the lessons learned from this exercise and provides recommendations on the choice of tools to facilitate the implementation of this method in other contexts;

II. CryoSat-2 PDGS Context

ESA's CryoSat mission is dedicated to measuring the thickness of polar sea ice and monitoring changes in the ice sheets that blanket Greenland and Antarctica.

A general description of the mission can be found in [2] and a description of the PDGS can be found in [3] (this paper references only information sources that are in the public domain; references to specific software vendors have also been omitted).

In summary, the measurements of the radar altimeter on-board the spacecraft are downloaded to the ground, where dedicated HW and SW infrastructure archives the raw data, processes them to create scientific products, and distributes them to the scientific community. The PDGS also provides the functionality to decide which measurement type to perform in which geographical area.

During the Transition stage, a set of documents was handed over to the O&M team. These documents included the standard set of technical documents: requirements documents, architectural design documents, Interface Control Documents (ICD), format specifications, and list of Configuration Items (CI).

The entire set of O&M processes is implemented as a Service, which is managed using an ISO-20000 [4] compliant Service Management Tool, which implements the Incident Management, Problem Management, Change Management, Release Management and Configuration Management processes.

The tool provides the capability to define a new CI and assign it to a class, chosen among a set of pre-defined classes. The tool also allows to define a relationship between two CIs, where the relationship name and type is chosen among a set of pre-defined relationships.

The tool does not provide the functionality to define new CI classes and new relationship types: this constraint represents the biggest challenge to overcome.

III. Description of the Process

Following good SE practices, the process has been broken down in the following steps:

1. Identify needs;
2. Derive requirements and constraints;
3. Define solution;
4. Test solution;
5. Implement changes;

The main need is to facilitate change impact analysis: when a change needs to be applied to a CI, it is necessary to assess the impact that it will have on other CIs. If this analysis is not performed, deploying the change could break one or more interfaces.

Without having the support of a SW tool, the change impact assessment needs to be performed manually, by collecting all the ICDs where the CI to be modified is involved, and determine if any other CI requires modification. This manual assessment is error prone and relies on the documentation to contain the correct, complete and up-to-date set of information required, which might not always be the case if the design documentation is not maintained by the O&M team.

The requirements derived from the need are:

1. Each component described in the architectural documentation should correspond to a CI in the CM database;
2. Each interface described in the architectural documentation should be implemented as a relationship between two CIs;
3. The CM tool shall provide the capability of selecting a CI and displaying all its relationships with other CIs;

In order to implement the requirement N.1, new CIs have been created:

- A set of CIs that describe external service providers (these were originally not present because their configuration control is performed by the service providers);
- A set of CIs that describe the fact that different instances of SW applications might have the same binary executables but different configurations;
- The CI class that was chosen to represent external service providers was "Data Center": this not an ideal name, but it was the closest match among the list of possible classes;

In the context of the CryoSat-2 PDGS, all interfaces are implemented, at SW level, as exchange of files.

In order to implement requirement n.2, it was necessary to take into account the fact that the CM tool does not allow using the same relationship name to connect different pairs of CI classes, so for example the relationship "interfaces with" can only be established between a pair of CI of class "Application", whereas the relationship "Exchange Data with" can only be established between a CI of type "server" and another CI of type "server" or "Data Center".

The CM tool used was already compliant with requirement n.3.

Figure 1 below shows an example of how the CM tool, after the implementation, allows to display all the interfaces of a specific CI.

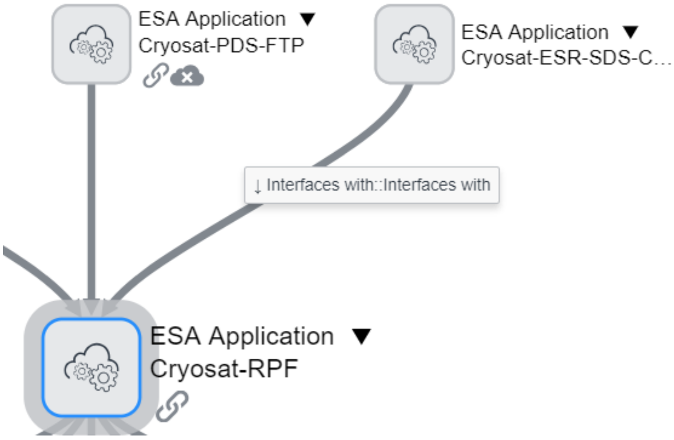

Fig. 1. Displaying the interfaces of a CI.

IV. Lessons Learned and Recommendations

The successful completion of this exercise has shown that it is possible to implement a mechanism that facilitates a robust change impact assessment by using only a CM tool.

In this example, the starting point was a System description captured in a set of documents, but if a System modelling tool is used, the model could be exported (e.g. as XMI [5]), or, if both the System modelling tool and the CM tool support it, an OSLC service [6] could be implemented that automatically creates CIs and relationships in the CM tool starting from the components and interfaces defined in the modelling tool. Another possible approach is to adopt the approach defined in the Modelling and Simulation High-Level Architecture (see [7] and [8]) to ensure interoperability between tools used in the Design phase and tools used in the Operations and Maintenance phase.

ACKNOWLEDGMENT

The author would like to thank the ESA staff at the ESRIN site for authorizing this exercise and provide the necessary resources to implement the recommended changes to the CM tool.

REFERENCES

[1] INCOSE, “Systems Engineering Handbook”, INCOSE-TP-2003-002-04

[2] https://www.esa.int/Our_Activities/Observing_the_Earth/CryoSat

[3] https://www.esa.int/Our_Activities/Observing_the_Earth/CryoSat/Data_flow

[4] ISO/IEC 20000 Information technology - Service management

[5] https://www.omg.org/spec/XMI/2.5.1/

[6] http://open-services.net/

[7] Möller, B., Garro, A., Falcone, A., Crues, E. Z., & Dexter, D. E. (2016, September). Promoting a-priori interoperability of HLA-based Simulations in the Space domain: the SISO Space Reference FOM initiative. In Proceedings of the 20th International Symposium on Distributed Simulation and Real-Time Applications (pp. 100-107). IEEE Press.

[8] Möller, B., Garro, A., Falcone, A., Crues, E. Z., & Dexter, D. E. (2017, October). On the execution control of HLA federations using the SISO space reference FOM. In Distributed Simulation and Real Time Applications (DS-RT), 2017 IEEE/ACM 21st International Symposium on (pp. 1-8). IEEE.