"SOFTWARE COMPONENT MODEL ALIGNMENT OF OSRA AND TASTE – CONCEPTUAL DESIGN"

Délia Cellarier [(1)], Régis de Ferluc [(1)]

(1) Thales Alenia Space France (Thales Alenia Space-F), France, E-mail:
delia.cellarier@thalesaleniaspace.com, regis.deferluc@thalesaleniaspace.com

Abstract: This paper presents the work that was performed by Thales Alenia Space-F, together with Bright Ascension and Viking Software, in order to define a harmonized approach relying on the alignment of two different software component models: OSRA and TASTE. The implementation concerns of this aligned component model were considered and led to some prototyping in the last version of TASTE, called SpaceCreator.

Keywords: OSRA, TASTE.

1. INTRODUCTION

The SAVOIR Onboard Software Reference Architecture (OSRA) [1] is a comprehensive reference architecture for spacecraft onboard (flight) software developed as part of the SAVOIR initiative. The associated component model (SCM for Space Component (meta-) Model) has been developed specifically for the needs of a spacecraft on-board software/flight software, that has to be capable of running on embedded targets with real time constraints, but also considering the specifics of monitoring and control.

TASTE [2] is a general-purpose modelling tool-chain developed by the European Space Agency (ESA) and dedicated to the software development of distributed and embedded systems. It is an open platform putting together the result of many years of research on MBSE techniques, including state machines (SDL), data modelling (ASN.1), architecture (AADL) and much more.

Unlike OSRA, TASTE does not address explicitly some important and specific design patterns that are required to comply to the current space system Standards (ECSS), such as those found in the Packet Utilization Standard (PUS), making it sub-optimal for an operational use.

The activity described in this paper aimed at defining an approach to align the two component models of OSRA and TASTE, with two objectives in mind: 1) to introduce OSRA concepts into TASTE (with priority given to concepts providing added value to users at short term, like monitoring & control); 2) to evolve the OSRA SCM model to allow modelling of the Execution Platform (seen as a black box in OSRA) with components.

For this, all features of both OSRA and TASTE component models were analyzed in order to determine if they were already aligned, if an alignment shall be performed or if it was not possible. The resulting harmonized approach is summarized in section 2. For each new feature that shall be introduced in TASTE, its impact on the new GUI of TASTE (SpaceCreator) was discussed. This is presented in section 3.

2. SW COMPONENT MODEL ALIGNMENT

2.1. Key concepts

First, the key concepts of the two component models were addressed: data types, component, interface, etc.

Even though the concepts of Component in OSRA and of Function in TASTE were not fully aligned, it was decided to map an OSRA Component to a root Function in TASTE. Then, it was decided to introduce some new features in TASTE, based on their specification in OSRA:

- The possibility to have several implementations per component;
- A new Interface concept, grouping operations and supporting inheritance;
- The definition of the Exceptions returned by operations (synchronous only);
- The representation of the Events exchanged between components.

On the contrary, other features of OSRA were not retained for the harmonized approach. It is the case for interface attributes and datasets.

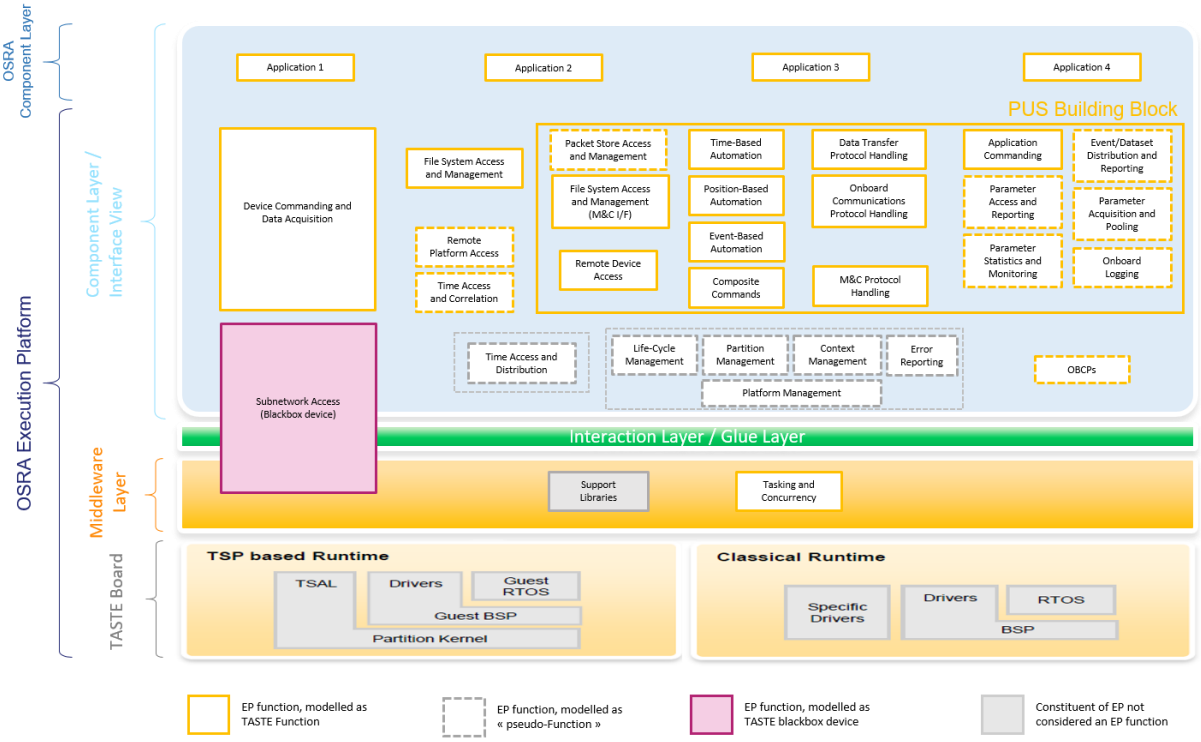

Figure 2-1: Repartition of the OSRA Execution Platform functions in the aligned architecture

2.2. Layered architecture

The OSRA component model only applies to the application layer, leaving the underlying Execution Platform as a black box, which interacts with Components through "pseudo-components". For the harmonized approach, the objective was to introduce components into the Execution Platform, giving the architecture depicted on Figure 2-1 . This figure shows that most of the functions of the Execution Platform are to be designed as components in the aligned component model.

In this new architecture, the pseudo-components as defined in OSRA are not needed anymore, except for functions of the Execution Platform depending on the underlying hardware or operating system (represented with dashed grey boxes on Figure 2-1). The implementation of these "pseudo-Functions" will not be provided by the user, but by the selected TASTE Board.

For on-board communication and device access, the device pseudo-component of OSRA disappears and instead, the SOIS layers are designed as Functions, using TASTE blackbox devices for subnetwork access.

Execution platform functions that represent the ground/board interface are gathered in a PUS building block. Another tool called OPUS, which is dedicated to the definition of PUS services, will be improved in order to generate this tailored PUS building block and the associated data types in TASTE.

2.3. Monitoring & Control

Monitoring & Control (M&C) services will be part of the PUS building block described above. The generated interfaces of the building block shall then be connected to interfaces of other Functions, but only for commandable operations.

Observabe/modifiable parameters can be defined in OPUS, or in TASTE as functional states of applicative Functions, tagged with M&C descriptors like in OSRA. To access those parameters, the usual approach with interface bindings will not be used. Instead, a kind of blackboard architecture will be generated by the tool, where data acquisition will be hidden in the middleware layer. A local API will be accessible by Functions owning parameters, and a global API tied to a data pool will be accessible by PUS services. Code generation will translate the use of local API to the global API. Local APIs will involve the use of functional states names, while the global API will use unique identifiers.

This M&C approach is illustrated on Figure 2-2.

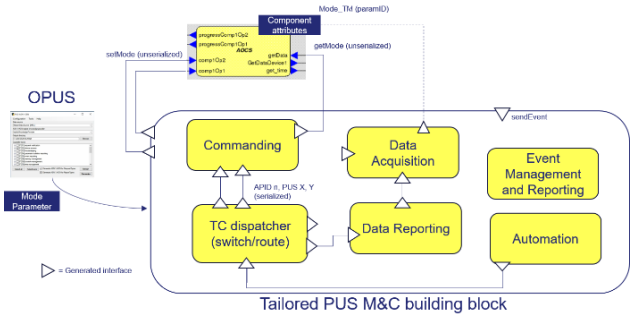

Figure 2-2: Example of PUS building block generated from OPUS

2.4. Dynamic Behaviour

Dynamic behaviour includes the following topics:

- Non-Functional Properties (NFPs) for Concurrency & Real-Time Behaviour;
- Interaction Patterns of operations;
- Tasking & Concurrency Model;
- Initialization of components;
- Sequence of Operations;
- Schedulability Analysis.

No new feature will be introduced in TASTE on these aspects, except the *bursty* operation kind.

Moreover, Message Sequence Charts (MSCs) in TASTE have been identified as a way to specify sequences of operations. A future model-checking engine could then check if MSCs are respected.

2.5. Time and Space Partitioning

The TASTE version from Ellidis is currently aligned with TSP aspects in OSRA thanks to the results of the MORA-TSP activity, although it identified room for improvements and future work. However, those aspects have not been implemented yet in SpaceCreator.

Concerning NFPs for dependability defined in OSRA, it was decided not to introduce them in TASTE.

2.6. Hardware Specification and Deployment

The concept of Board in SpaceCreator is close to the Processor Board entity in OSRA SCM. For now, the aligned component model will not include other entities supported by OSRA (Devices, RTUs, Network Switches…). This could evolve depending on the results of an on-going study which aims at merging COMPASS [3] and TASTE.

2.7. Applying the SW component model

Topics related to workflow, process, platform configuration and link with a Spacecraft Database (SDB) were also discussed.

In the end, it was specified that there should be a hook to get values from an external source during code generation. For example, APIDs either come from the SDB or from OPUS tailoring, that will then generate a PUS building block with context parameters. If the SDB contains all default values, and without OPUS in the process flow, the SDB can come to overwrite TASTE default values during code generation.

3. UPDATE OF THE TASTE GUI

The implementation of the new features coming from the alignment described in section 2 will have impacts on the GUI of TASTE. For each feature, an update of SpaceCreator (the new GUI for TASTE) was specified in this activity.

Some of those updates could be prototyped during this activity: the support for multiple Function implementations, and the support for "pseudo-Functions".

Figure 3-1 shows that a new *Implementations* tab was introduced in SpaceCreator, gathering all the implementations of a Function. A new implementation language is also available to specify if it is a "pseudo-Function".

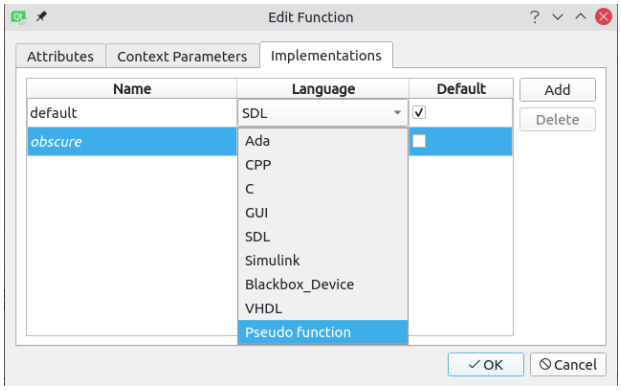

Figure 3-1: New implementation tab in SpaceCreator, with the "pseudo function" implementation language

In the Deployment View of SpaceCreator, the user can specify for each deployed Function which implementation shall be used. In addition, a Board now provides information about the "pseudo-Functions" it provides.

4. CONCLUSION

The harmonized approach defined in this activity gives the specification of new features to be implemented in TASTE based on the OSRA. Those new features will make TASTE more suitable for on-board software development for space missions, by better assisting users in the design of ECSS-compliant software. A few features were prototyped in SpaceCreator, but future work shall be conducted in order to implement the complete harmonized approach, including further developments for the OPUS tool. This shall allow OPUS and TASTE to be operationally used on future ESA missions.

5. REFERENCES

[1] "OSRA- Onboard Software Reference Architecture," [Online]. Available: https://essr.esa.int/project/osra-onboard-software-reference-architecture.

[2] "TASTE - A tool-chain targeting heterogeneous embedded systems, using a model-based development approach," [Online]. Available: https://taste.tools/.

[3] "COMPASS," [Online]. Available: https://essr.esa.int/project/compass .