Component Model / Meta-Model Reference Implementation

Presenter: Marco Panunzio (TAS)

The SAVOIR-FAIRE On-Board Software Reference Architecture (OSRA) comprises three architectural layers: the component layer, the interaction layer, and the execution platform.

Components developed for the OSRA are specified according to a "component model", which defines: (i) an overall methodology that establishes the concerns to be addressed by component (and those that are by choice left to external definition), the steps necessary for component specification, and the relationship of those steps with exchanges outside the software development team (e.g., requirements, an export from the SDB); (ii) the range of properties that individual components must satisfy; (ii) methods, and possibly mechanisms, for composing components.

The methodology underpinning the OSRA and its component model is centered on the application of the separation of concerns principle.

This is reflected in: (i) the design space and component definition, which clearly promotes and enforces separate specification of functional and non-functional concerns (i.e., timing, synchronization, tasking and space); (ii) in the implementation space, where distinct software entities are in charge of distinct concerns (i.e., *components* for functional implementation; *containers*, for the realization of timing, synchronization and tasking concerns; *connectors*, for realization of interaction between components).

In this presentation we present an overview of the OSRA component model and its main design entities. We also highlight what software concepts are included in the OBSW design model (termed "core model") and what concepts are left to external definition, either on models / artifacts under control of the development team, or provided by other satellite teams (e.g., SDB, data handling).

Finally we present the Space Component Model (SCM), which is a reference implementation of the OSRA component model.

SCM is defined as a set of domain-specific metamodels, by which it is possible to create a model of the OBSW conforming to the OSRA component model.

We present the main drivers for the reference implementation, the language architecture and the advantages in terms of design correctness for the overall approach.