Bridging the Gap between On Board and Ground Configuration Data Bases

Flying an Onboard Software (OBSW) is not just flashing the EEPROM. The binary image has to be delivered together with source code and memory map, but also with many software artifacts such as User and Operation Manual (UOM), Interface Control Documents (ICDs), Test Environments and Satellite Reference Database (SRDB) for configuring the Ground Facilities. Most of these artifacts are developed independently from each other, the coherency of the complete set being verified through long reviews and exhaustive integrated tests that comes at a late stage in the process.

A more efficient way is to ensure coherency by deriving all these products from a single source of truth based on a single formalism, through validated chains of model transformations. Some initiatives have shown the benefits of such approaches. These are however for now mainly limited to flight code, test environment and interface document. It is proposed to extend them to the configuration Ground Facilities.

The PUS-C Gen study has for instance demonstrated that a Generic PUS-C model could be customized and extended for a specific mission. The result of this customization is, amongst others, an ASN.1 definition of the Space-Ground interface. From this definition, the ASN1SCC compiler is able to generate encoders and decoders for the OBSW and for the test environment, but also the ICD in HTML representation.

At the other end of the communication link lies the Ground Facility, with legacy products such as SCOS2000 or more recent ones such as the EGS-CC. In order to communicate with the Space Segment, the Ground Segment has to be configured with the very same TM/TC description as the Space Segment. As today, there is however no direct formal link between the ASN.1 definition and the SCOS2000 Database. These are two different kind of formalisms. On one hand, we have a textual definition supported by a Domain Specific Language. On the other hand, we have a pure Relational Database schema. Is it possible to bridge the gap?

The proposed solution is to capture a decorated version of the ASN1SCC Abstract Syntax Tree (AST) in a model that contains all the information regarding the TM/TC structure and representation. This information is the as the one used to generate the encoders and decoders software and the interface documents. However, the ASN.1 description only covers the structure and representation while the SRDB contains additional information such as calibrations, ground monitoring and display information. The AST model has therefore to be integrated in a larger model that allows capturing this additional information while, for each TM/TC, it shall refer to the AST elements, including fields data type, size and (variable) offset.

A chain of model transformations can then lower the representation into a relational database. During the transformation, all the intermediate models are verified according to constraints and mission specific naming conventions are generated. The user's comment in the PUS-C Gen ASN1 TM/TC definition can even be automatically flown down into the description of the corresponding field in the SCOS2000 display.

The processing involves model to model transformation through three levels of models: Domains Models, a single Unified Model and Implementations Models. Such a chain of processing is common inside a single tool. What is introduced here is the Unified Model that leverage tools interoperability. The intend of this Unified Model is not to have again one single universal (and complex) model that suggest/enforce a combination of notations and tools. It is to have a light modeling environment allowing small teams to customize the tools they already master to their needs.

Domains models are specific to one aspect of a system (system decomposition and interfaces, behavior, data representation,…).

Front-end tools first analyze these Domain Specific Models, providing their contributions to the customized Unified Model. These tools are Graphical or Textual DSL Editors. Most of them are

supported by ‘Solvers’ that generate a solution according to domain constraints. Examples of such tools are: DSL compilers for ASN1, SDL, AADL languages or modeling environment for UML, Matlab, Simulink

The Unified model then puts in relation and extend the output from the previous processing in a common and general purpose modeling environment based on interchange XML files. Cross model validations are then performed.

This Unified model is then lowered in different Implementation models using M2M transformations. Software artefacts are then emitted from Implementation models using final M2T transformations. Examples of such implementation artefacts are: SW or HW configuration tables, technical documentation, relational databases, inter-operability interfaces or FPGA bitfiles, …

With this approach, the SRDB content is, by construction, congruent with the OBSW. All this information is generated from a single high-level source of information, through a trustable chain, reducing system integration and validation effort.

Dominique TORETTE
SPACEBEL