SESP 2015

Conceptual data model utilisation in EGS-CC

Harald Eisenmann(1), Anthony Walsh (2)

(1) Airbus Defence & Space
D-88039 Friedrichshafen
Germany
Email: Harald.Eisenmann@airbus.com

(2) European Space Agency ESOC
D-64293 Darmstadt
Germany
Email: Anthony.Walsh@esa.int

Data Exchange is an important element in the overall EGS-CC context, to ease the collaboration in European Space Programs. The data model engineering was considered an enabling task being part of the System Engineering Team activities. A dedicated methodology has been defined, relying on proven mainstream technologies. This methodology was supported with a dedicated tool set supporting data model engineering and validation. Related extractors of the CDM to utilize the CDM for the S/W development are either in progress or being planned. With this the EGS-CC CDM is not only a definition of the data, but it is actually being used for the S/W development and cares for improved consistency.

NEED FOR DATA MODELING IN EGS-CC

EGS-CC is a European Initiative to jointly develop core Monitoring & Control elements for use in AIT and operations, in order to reduce cost for share and exchange of information in the process, allowing the re-use of data between different activities and sharing common elements in general. In order to achieve this, a common understanding of the data to be managed along the supported processes is mandatory. Moreover this understanding requires a clear definition of the semantics (meaning) of the data, since this is typically essential if sharing, exchanging and re-using data is to be achieved. To specify the semantics, modelling languages for data need to be used. Having a data model available, might play a major role in the EGS-CC development process, where it comes to provide efficient interfaces between preparation environments (e.g. a system database), and CCS / MCS, or between (different) preparation environments of different organisations. Modern S/W development techniques allow the automated utilization to ease development and verification costs.

APPLICATION OF DATA MODELING

Data modelling is a relatively young activity in support of space engineering. Typically the data modelling is covered as part of the S/W development process, very close and very dependent on the actual S/W development process and applied technologies. In that it had little exposure to the end user. However it is a common understanding in European space community, that for overcoming the issues on data exchange, a conceptual data model, established prior to the actual S/W development can play a major role. This conceptual data model focuses more on semantic aspects and less on S/W development issues, like technologies or performance. In that the CDM is a very valueable resource from for the user community since the information / concepts are addressed on an application domain level. For example a telemetry packet can be decomposed into the elements from which it is formed of, i.e. header, footer, parameters or calibration curves. All of this is specified at the conceptual data model together with the required consistency checks to ensure integrity and completeness.

Since the activity is young processes and technologies are not yet consolidated and shared. For the technologies and tools different streams are observable. In order to establish the approach the following elements have been considered with respect to its readiness and availability but considering the overall end-2-end process. From there a method has been derived for EGS-CC which can be seen as consolidated best practices, but give clear emphasis to the possibility to have close and proven interface to the S/W development, for enabling a smooth utilization of the CDM there. The process covers the following activities:

- Analysis of the application domain: Here the main focus is to understand the current existing models, specific needs, but also potential short comings in the current solution. In order to ease the discussion, no particular format has been requested, slides or spreadsheets where rather appreciated to clarify and share the concepts
- CDM specification: In order to have a clear baseline of the scope and content, the CDM was specified first with text based requirements. The intention here was to specify the data from a user point of view, meaning

how the data appears at runtime (at instance level). The requirements have been expressed as text / spreadsheets.

- CDM definition: Based on the CDM level requirements the actual CDM has been defined. Initially UML has been used, in order to keep the link into the S/W IDE. This decision has been revised and Ecore was used as language to express the CDM. The decision for Ecore was due to the fact that the Ecore meta model is very close to the concepts used from UML, but having not the burden of many elements not needed.
- Pre-validation: The pre-validation is a step, which is to be performed, before the actual S/W development process is being started. The main objective is to mature the CDM, by the instantiation of selected examples of the CDM. This effort can be seen as a risk mitigation effort, in order to reduce the number of changes of the CDM during the S/W development (although they cannot be completely avoided). In order to reach this goal a representative set of samples is to be applied during the pre-validation.
- S/W development: In order to benefit of a formally specified CDM it has to be applied during the S/W development. There is no specific issue related to the CDM utilization, since mainstream technologies are applied in EGS-CC, which nowadays do all rely on model driven S/W engineering, where the CDM approach in EGS-CC perfectly fits into that. A key element during the S/W development is the model maintenance. While on one hand the formally correct specification is a quite valuable input to the development, the model (even if it would be 100% correct), but by subject for a change. The request for changes will be the result of S/W development issues, either due to the technologies applied (e.g. that some constructs cannot be expressed in the implementation domain, or one with a slightly change) or that the model is to be updated for performance reasons. If this is required, in EGS-CC the CDM would remain the same, but the CDM would be patched along the automated model transformation from the CDM into the S/W IDE.
- Validation: The final, and exhaustive validation of the CDM, can only be performed in the final target environment. Here an extensive and exhaustive validation of the CDM is taking place. Most likely the model will be validated implicitly with the validation of the overall system. But is also possible, to re-run specific representative cases from the pre-validation.

TAILORED SOLUTIONS IN SUPPORT OF APPLIED METHODOLOGY

In EGS-CC it has been decided to apply a tailored tool set supporting the data model engineering process. The tool supports all activities from requirements specification through pre-validation, including requirements definition, data model definition, definition of validation cases, and also the instantiation of validation data sets.

This tailored tool was derived from a modelling framework developed in the frame of ESA TRP, relying on the Eclipse modelling framework, which allowed configuration according to the CDM engineering process. There was a trade being performed, using a COTS UML tool, which was the initial baseline in EGS-CC. Basically the main driver was complete support of all activities in an integrated framework, fidelity of the provided features and also the ability to tailor the language being used for the data model definition.

The tool-set is composed on 2 different, individual tools. One is used for all activities on CDM engineering, on the “meta-level”. This includes the requirements definition, CDM definition and definition of validation cases. The tool is called Data Model Editor (DME). The DME comes with different editors supporting the definition of the various aspects. Basically the editors allow tree-based instantiation and navigation, and tabular entry of data, UML class diagram like definition of the CDM and also and “exploration editor”. This exploration editor provides a graphical retrieval of the actual model content. This is in particular well suited, if the model is visited with specific questions, or to get familiar with the model. The graphical views are continuously layouted automatically, the queries can be persisted. With a specific view of the model can be re-visited, the resulting diagram will be automatically created, considering the actual model status.

The second tool being used is supporting the instance level activities, the CDM pre-validation, called Instance Model Editor (IME). This tool is used to perform the pre-validation activities. For this the IME provides the mean to create instance level examples of the CDM. The IME supports this with graphical views similar to the DME, tree-, table- and explorer based instantiation of the CDM. In terms of the IME tool architecture, the following is worthwhile to be mentioned: The tool set-up can be distinguished into 2 parts, one is a generic part, which is providing the basic functionality such as model-management, persistency or the mentioned editors. This generic part is independent on the actual CDM managed. The specific part provides the containers in order to instantiate the CDM. Basically it represents the classes which are necessary for this purpose. In that the specific part of the IME is directly depending on the CDM. Moreover the specific part can be generated from the CDM. The generic part however, since it is independent on the CDM, can be used for “any” data to be managed. In that it can also be used to support the DME definition. With that a significant part of the IME can be shared with the DME.

As mentioned that DME / IME rely on a small data management framework, which is a spin-off, from ESA TRP. Relying on this did not only enable to build a small, agile environment, which eases to follow the evolving CDM definition. In addition to that some specific features have been added to the DME / IDE, dedicated to data modeling, which are not available in other tools. Some of the features are worthwhile being mentioned here:

- Categories: In the CDM a careful balanced decision had to be taken between an explicitly modelled CDM, where all classes and properties are defined, and a CDM which only provides a very generic skeleton. The basic constraint behind is, that on one hand, an exhaustive definition of the CDM is needed, on the other hand having all data “hard-wired” in the CDM, requires that the S/W will have to be changed for each property added. In EGS-CC the CDM offers to define “Categories”. Categories can be seen as *lightweight* classes which can be defined at runtime. The only constraint is that Categories, always have to be “attached” to and instance a *real* class, which was defined in the CDM before. Example: The CDM introduces SystemElements, key building blocks of the system. This does not make any reference to a particular type of equipment. With the categories, a system element can be decorated with categories and properties, e.g. for the performance (e.g. startracker performance), geometrical accommodation, or validation status. Categories are a powerful, but very tailored specific concept in EGS-CC. The DME / IME provide the means to proper support the category definition. In the DME categories, can be defined, together with the CDM definition. In the IME it is possible to load the category definitions which have been defined before in the DME, and further evolve them. In addition to that the IME also allows to “instantiate” the category defintions, meaning to actually define related property values.
- Explorer: As a above mentioned the DME / IME provide an interactive editor, which allows to traverse the model through queries. The structure for the queries is defined in the meta level, the language which is being used for the corresponding definition. With that e.g. in the DME the queries are defined based on Ecore, in the IME the language being used is the CDM of EGS-CC.
- Constraint definition and validation: The DME full supports the complete constraint definition. In the DME a dedicated editor has been added allowing the definition of OCL constraints, based on the OCL meta model. For the OCL definition the CDM is referenced the proper references are managed as part of the OCL definition. This eases the definition, e.g. through auto completion or search for constraints which are incomplete, since the CDM has been changed.
- The DME allows to define the export of various artefacts, which are strongly dependent on the CDM, in order to have a consistent flow of information from the CDM into the implementation. This allows to effectively use the CDM for the development of EGS-CC

UTILIZING THE CDM FOR THE DEVELOMENT

The definition of the data to be management in a S/W system, the data model, is a crucial definition for any development. This is because the data managed, will occur in different places in the S/W, often relying on different implementation technologies. Examples for this are e.g. the MMI, which provides the graphical exposure of the data to the user, internal data management system, database, allowing the corresponding instantiation and management, persistency, and archive, management of importing and exporting data, …, and many more. The data need to occur consistently in the different views in a S/W system. If this is managed manually it can be a labour intense and error prone activity. Model driven S/W engineering aims to utilize formally specified (specification) models in order to derive the data for code generation. This is a key driver for managing the costs of the maintenance phase. In this the CDM can be seen as specification of the S/W.

In EGS-CC the following artefacts have been identified which are directly depending on the CDM:

- Instance Model Editor: As described above the IME can be considered a light weight database allowing the CDM instantiation for validation purposes. Part of it is directly depending on the CDM and can be automatically generated from it.
- S/W Design: The S/W design has to clearly address the data managed in order to specify the relevant runtime data structures.
- Data Exchange Schema: Data exchange is one of the key topics addressed in EGS-CC. Since the CDM defines the data managed in the scope of the EGS-CC supported process, it sets the scope for the data which is to be exchanged. From there it is obvious to utilize the CDM to generate the relevant exchange schema.
- API: Having a data exchange schema established an API is needed in order to access the data which is to be exchanged. Obviously also the API has to address the CDM defined data. From there it makes sense to consider this as automated generation.
- CDM Report: A CDM report is needed as reference to the defined CDM.

For all automated derivations of data a dedicated process has to be established. This process needs to be defined in terms of actual technical data model transformation, but also in stakeholder ship. For it is important to consider the relevant model abstractions. In S/W engineering typically 3 different abstraction levels can be distinguished, which are also tangible in data modelling:

- Conceptual Data Model: It is an abstract view on the data conceptual (data) model. For S/W engineering this model is also called domain or architectural view. This view focuses on the actual needs and less stresses the constraints resulting in the application of particular technologies. Basically the conceptual data model is a formalized model of the problem space.

- Logical Data Model: This is a more elaborated view on the data considering the constraints coming from implementation. Those constraints can be distinguished in the plain technological consideration of technologies/languages (e.g. SQL, Java, XML) and also in the specific implementation issues of the particular application, in particular performance. Both might impact the LDM, with respect to the CDM definitions. For the technologies, this might result in changes since there is not always a 1:1 mapping of language constructs possible. Considering implementation issues, might even result in altering the CDM.
- Physical Data Model: This is the actual realized data modelling in the target language, the source code.

For EGS-CC in terms of governance of the different models the following is of importance. The CDM shall be the unique, reliable reference of the data model definitions for all derived artefacts. Since the CDM specifies the end user concepts in the model, the CDM will be managed by the SET. For the different LDMs, and there might be different LDMs, in principle per derived artefact, the stakeholder ship is different. For example for the S/W related LDMs, clearly the stakeholder will be the S/W development team. For any issue with LDM, it needs to be clarified, if the LDM – or later on the PDM the implemented data model, has flaws, because of the issues in the CDM, or that the issues are more related due to technology or implementation issues and constraints. For this a process is to be established discussing the related issues, and to agree, whether the issues are implementation / technology related, or rather that the root cause is the CDM. In the latter case the CDM needs to be updated. If not then a “patch” to the LDM needs to be established. This patch can be implemented in various ways. Either it is a hard-code modification of the CDM during the transformation, that this patch is actually applied to the CDM during the transformation, or that the LDM is manually modified. In this case the modification has to be done persistently, which means with every new generation of the LDM from the CDM, those changes shall be preserved. Independently on the particular implementation is crucial for the actual CDM utilization that, all CDM related issues are discussed with the stakeholders involved / affected, and that all modifications of the LDM vs the CDM are commonly agreed and completely traceable.

Besides the flow of the CDM into the S/W design the utilization of the CDM for data exchange seems the most critical use case – for EGS-CC. The CDM needs to be considered in various different ways. First of all there is the data exchange schema, which is to be derived from the CDM. Furthermore an API is needed in order to access the data which being exchanged. It is a commonly used practice for data exchange to provide an API on “business object level”, which is more or less the domain level as specified in the CDM. Typically for this purpose the business level objects are represented as “Plain Old Java Objects” POJO, in order to provide simple access to the exchanged data. It is obvious that the API can be generated from the CDM. However for the actual data exchange, however a low level access on XML level is needed in order to convert the XML exchanged data into the CDM level API. There are many XML level technologies available providing this feature. Important is that with the utilization of the CDM the data exchange solution can be configured automatically, without major manual intervention.

For the data exchange in EGS-CC a specific aspect has to be considered. It is the use of XTCE for particular use cases, i.e. the exchange of data from and to systems outside of the scope of EGS-CC. The XTCE utilization requires a dedicated manual tailoring of XTCE. For this tailored XTCE the relevant import / export routines reading writing XTCE files have to be also implemented manually. With this they form a self-standing XTCE API which can be shared. The integration of it requires the consideration of the dedicated tailoring.

For the use of XTCE in the EGS-CC context, however this tailoring needs to be mapped to the EGS-CC CDM. This mapping can also be used to develop a dedicated mapping layer from the XTCE API, to the CDM level API. With this approach, both the, CDM level, native data exchange and the XTCE based data exchange, can be integrated into the CDM level API. This eases the use of the data exchange solution significantly since it is independently on the actual format being used for the exchange.

CONCLUSION

From the very beginning in EGS-CC the data being managed in EGS-CC the data model has been considered a very important item, to overcome data exchange solutions. The approach defined was taking into account the various stream of data modeling available, from ontologies, fact based modeling to more production oriented solutions like Ecore or UML. Priority has been given to production oriented solutions, relying on mainstream technologies. But the approach taken for data engineering accommodates some elements from the other streams in a pragmatic way. This approach was supported with a light-weight tool set. With this EGS-CC is well prepared to support the development and maintenance with a stare of the art solution.