# SRDB Next Generation

Harald Eisenmann[1], Claude Cazenave[2]

*[1] ASTRIUM Satellites*
*D-88039 Friedrichshafen*
*Email: harald.eisenmann@astrium.eads.net*

*[2]ASTRIUM Satellites*
*31 rue des Cosmonautes*
*Z.I. du Palays*
*F-31402 Toulouse Cedex 4*
*Email: claude.cazenaven@astrium.eads.net*

## INTRODUCTION

The project's Satellite Reference DataBase (SRDB) hosts all data that needs to be shared along the project life-cycle, in order to define and test the avionics and the operations of a spacecraft. It includes various kinds of data used by various spacecraft domains like e.g. :

- TM/TC and operations data (including ground segments)
- Data for the production of the central software
- Electrical information at functional level
- System and environment data for the configuration of the simulation facilities

Astrium is currently preparing for the development of a new SRDB, using state of the art technologies. This papers reports on the approach planned and outlines some details on functionality, interfaces and architecture. The structure of the paper is as of the following:

- Introduction of the context where an SRDB will be operated
- Identification of high level functionality provided by an SRDB
- Identification of key technologies and architectural outline
- Relation to the ECSS E-10-23A Engineering Database
- Brief summary of validation activities performed

## CONTEXT

Historically these kind of of systems have been called system or satellite system databases. The role of the classic system databases have been evolved towards 'reference databaseses', emphasizing the aspect of data which needs to be effectively exchanged, shared and provided ('one data one source'). SRDB is to provide a reliable data source repository with strict configuration control processes. Beyond the classic SRDB use case (avionics development and validation) over the last decade the use cases evolved - along with the trend of model-based systems engineering (MBSE). MBSE demands that increasingly more and more data is formalized and moved from documents to databases.

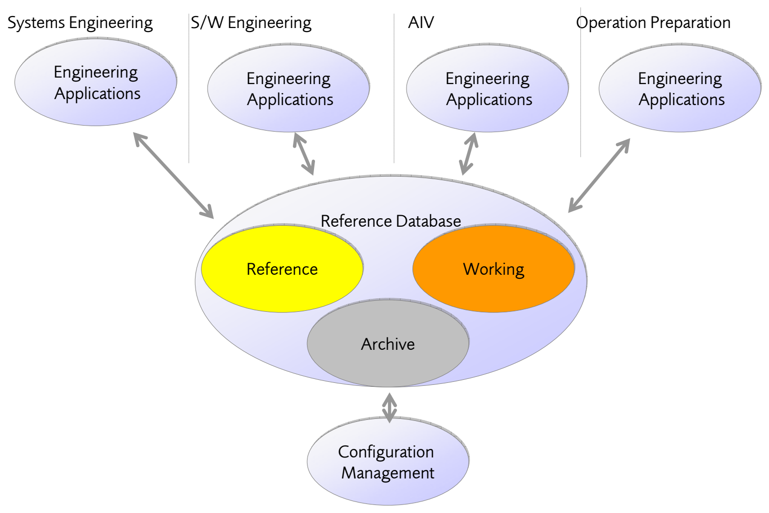

**Figure 1: SRDB context**

By using modern S/W technologies the reference database can be considered as the central information hub collecting and distributing data – on a semantic correct level – between different engineering disciplines, sites and companies - for concurrently performed engineering processes and last but not least along the customer supplier chain.

The data management provided can be considered as centralized services shared between the different engineering applications ('authoring') systems. Thus the functionality provided is complementary to the traditional PDM systems providing essential functionality for the overall project configuration control.

Figure 1shows the context of the classical SRDB to support avionics related activities. At a given time processes related for systems engineering, S/W engineering, AIV and operation preparation are concurrently performed. Therefore SRDB systems typically internally have different areas, where data still can be edited ('working'), or is serving as reference to start new activities ('reference') or is kept for consistency reasons ('archive').

**FUNCTIONAL SCOPE**
From a reference architecture perspective the any engineering application (e.g. design tool, analysis tool, functional simulator, …) can be decomposed into the following typical functions.

- Man-Machine-Interface, providing an interface for interaction with the end-user. Typically this is a graphical user interface allowing the use and operation of the engineering application.
- Core Application: This is the key functionality for the application, and provides either data management, simulation, or e.g. editing functions.
- Data Persistence: This provides the permanent storage of data, exceeding the duration of a session. The technologies used cover a whole range

Figure 2 shows a sample set of different tools, which have to be integrated. The red dashed rectangle identifies the scope of engineering database as identified in ECSS E-TM-10-23. It comprises all the data handled in the different tools which is to be shared, exchanged, managed (e.g. in terms of consistency check, history tracking, …) and stored. E-10-23 is furthermore asking for a semantic integration of the different tools. This is important as the different tools do not only differ in terms of implementation technologies for data persistence and data management, but also the underlying data formats – the meta-models. The semantic integration will "transform" the data on the same representation, in order to ensure a consistent management of data.

This is the role of the future SRDB, and thus it can be considered as an engineering application of its own. It falls into the same basic pattern. The SRDB provides commonly shared functionality for data persistence, data management, semantic integration and data exchange. In order to obtain a set of tools which to interoperate in a correct fashion it is essential that the data management functionalities are centralized.

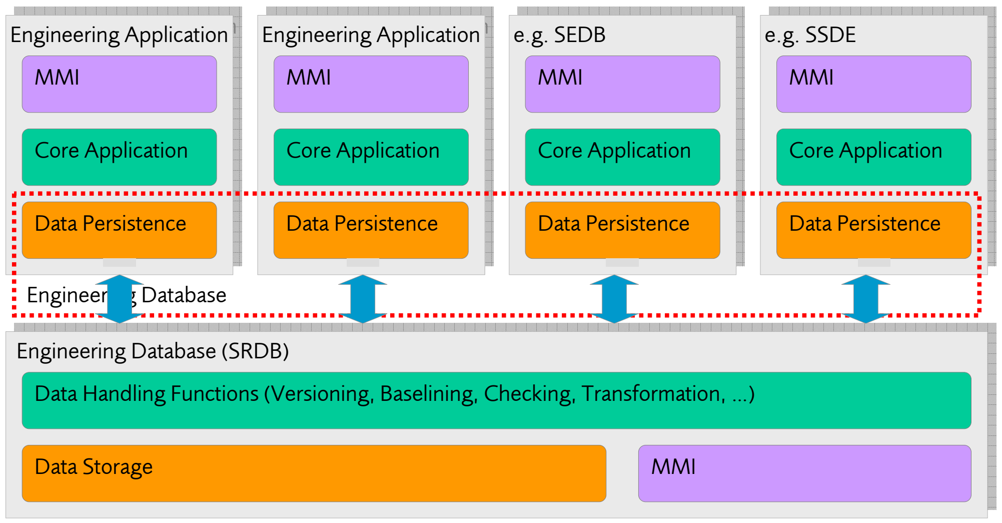

**Figure 2: Functionality Engineering Application**

The functional decomposition is shown in Figure 3, it comprises the following functions:

- Data Management is the core function set to handle the actual stored data in terms of consistency checks, persistency, history tracking or the synchronization for concurrent data access.
- Data View provides a graphical view on the data e.g. in terms for tree, table or property sheet view. The views can be either for viewing or in a full editable fashion.
- The data import export is to exchange / deliver data between the different user / provider of the data

- For conversion or transformation of data from / to other representation dedicated adaptor functions are used in combination with the import export functions

A specific set of functions deal with the data model. In current systems often enough the data model is tightly linked to the implementation. This does not allow an evolution of the technologies neither the data model itself. By using state of the art S/W engineering technologies (Model-driven architecture) it is possible to define the data model independently from the implementation technologies. The associated functions are:

- Data Modeling: This function allows to define the data model. The actual tools and technologies used for the data model definition (e.g. Express, UML, ORM, OWL, …) are less important then the fact of expressiveness and implementation-technology independence.
- Transformation: In order derive the required code from the data model, by considering the constraints from the technologies used.

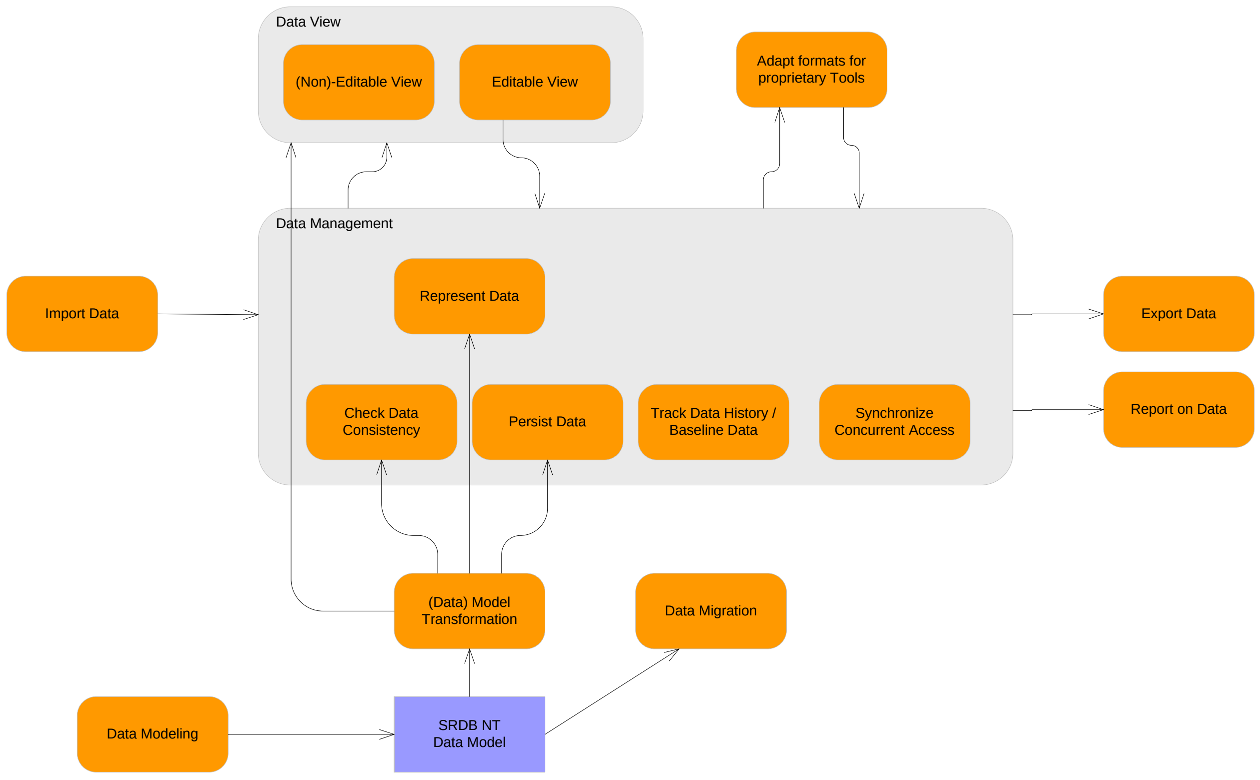

**Figure 3: Functional Decomposition SRDB**

## TECHNOLOGIES AND ARCHITECTURE

Over the last years the technologies for implementing databases and data management evolved a lot. In the past decades relational database management systems typically have been used for this kind of problems. The recent emerging technologies however, do have a massive impact on development- and deployment approach. Some key technologies are the following:

- Model-driven-Architecture (MDA): Considering the innovation cycle for S/W implementation technologies, the awareness for the need of more abstract models increased. Those models are to decrease S/W development costs through automated code generation. For database development the key model driving the development is a common –conceptual- data model. If it is managed to keep the data model independent of implementation technology aspects, a common model can be used to derive data structures for different implementation technologies e.g. among others XML, SQL or Java. Thus applying MDA has a massive impact on the S/W development approach and fully pays of in terms of reducing development and maintenance costs, reduced development time and openness towards technology evolution.

- Service-Oriented-Architecture: Besides the improvements for the S/W development process the other massive impact is on the “deployment” architecture. In the past – even in Client / Server applications – the different tools have been tightly coupled. The different functions provided by the different tools could be shared by using protocols like e.g. RPC or CORBA. This lead to a “hard-wired” deployment architecture of tools. In particular if it comes to process support, where the tools have to be adapted to meet the process requirements.

Service orientation in principle introduces the distinction of service and function. An application server allows the hosting of services. The service users just have to access the application server with the correct interface. The implementation of the function hosted on the application server as service is hidden. This provides more flexibility and modularity for the deployed architecture in terms of transparency for access and location of the actual function provided as service.

- Eclipse: A further interesting implementation technology results form the open source community initially aiming to support the S/W engineering process by providing a free S/W development environment (SDE). Based on the open, robust framework Eclipse evolved into a powerful library to build MMI's, further into a full fledged data management core. Figure 4 gives an overview on the relevant parts of the eclipse framework. The core part of it is EMF.edit and EMF.model. This allows the representation of the data in away allowing the direct manipulation of the data. Basically EMF.model represents the data itself, while EMF.edit provides the interfaces to manipulate the data. On top of this core, graphical editors can be build. Tables, trees, property sheets can be developed by using the means of EMF. On top of it Eclipse GMF provides the resources to develop graphical diagram oriented editors. The whole Eclipse framework is realized in Java.

Based on this EMF core further data management functionalities are provided. Those comprise e.g.:

- Compare: for the comparison of different EMF data sets
- Merge: for merging different EMF data setst
- CDO: for allowing and synchronizing concurrent access
- Ecore: for the specification of the data model, which used to generate the EMF.edit and EMF.model classes in the second step
- Xpand: for the definition of data transformation rules
- XMI: for a simple model serialization and persistence
- Hibernate: for the object-relational mapping for a persistence using a RDBMS

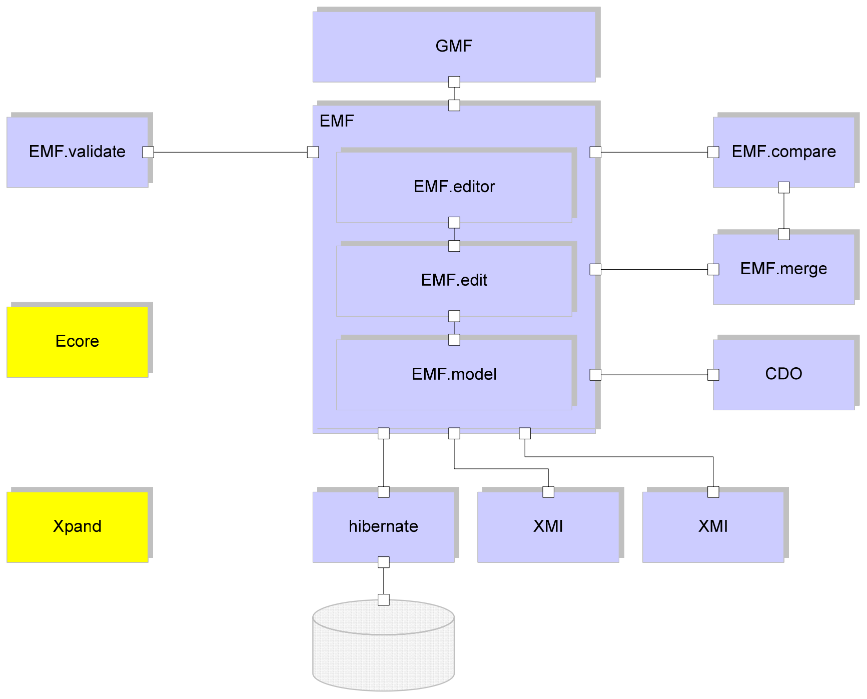

**Figure 4: Eclipse Framework**

Not shown is that is Equinox. Equinox is the Eclipse implementation of the OSGi (Open Service Gateway Initiative) specification for an open Java based service framework.

Figure 5 shows the envisaged architecture for the SRDB NG, resulting on the considered use of Eclipse. The scope of SRDB NG covers the data management entirely running on an OSGi application server and the "Data Editor". Besides the tools belonging to SRDB NG it is also possible to integrate 3rd party tools.

The DataEditor is the end user application allowing the visualization and manipulation of data. Each user will have its own session of a DataEditor running. It is foreseen to investigate in a rich-client where a data set is exclusively locked by a user and transferred to a users computer (for server independence reasons e.g. performance, campaign, …) and a 'thin client' where the data remains on the server where additionally concurrent access can be provided.

The data management itself hosted on the application server is organized to operate on data sets. DataSets can be versioned and organized in a graph with branches, where branches can be created (‘split’) or merged. The functionality for the data management comprises

- SemanticDataChecking: Apply a selectable set of constraints on a data set. The checks might be expressed in Java or OCL.
- ConcurrentDataAccess: Make a data set available for concurrent access from different client sessions at the same time.
- Comparison: Provide the comparison of 2 or more data sets, e.g. in different versions or branches. Also a merging  of 2 data sets can be performed.
- Baselining: The baseline allows to freeze a complete data set.
- Hibernate: Provides the object-relational mapping in case a relational database is used as underlying persistence.

The persistency ensures the availability of data in case of a shutdown or crash of the application server, or synchronization with external systems. It is foreseen to investigate into the following:

- Relational Database Management System: From an RDBMS only the persistence functionality would be used as the data management is realized entirely in Eclipse and hosted on the application server.
- File: The simplest solution to achieve persistence would be using XMI, which is automatically integrated in Eclipse.
- VersionControl: For this e.g. SVN or Git. A combination with SVN or Git/XMI could provide an easy to achieve archiving and versioning
- PDM I/F: An interface to the EADS developed PDM system is important to have the SRDB integrated with the overall configuration control.

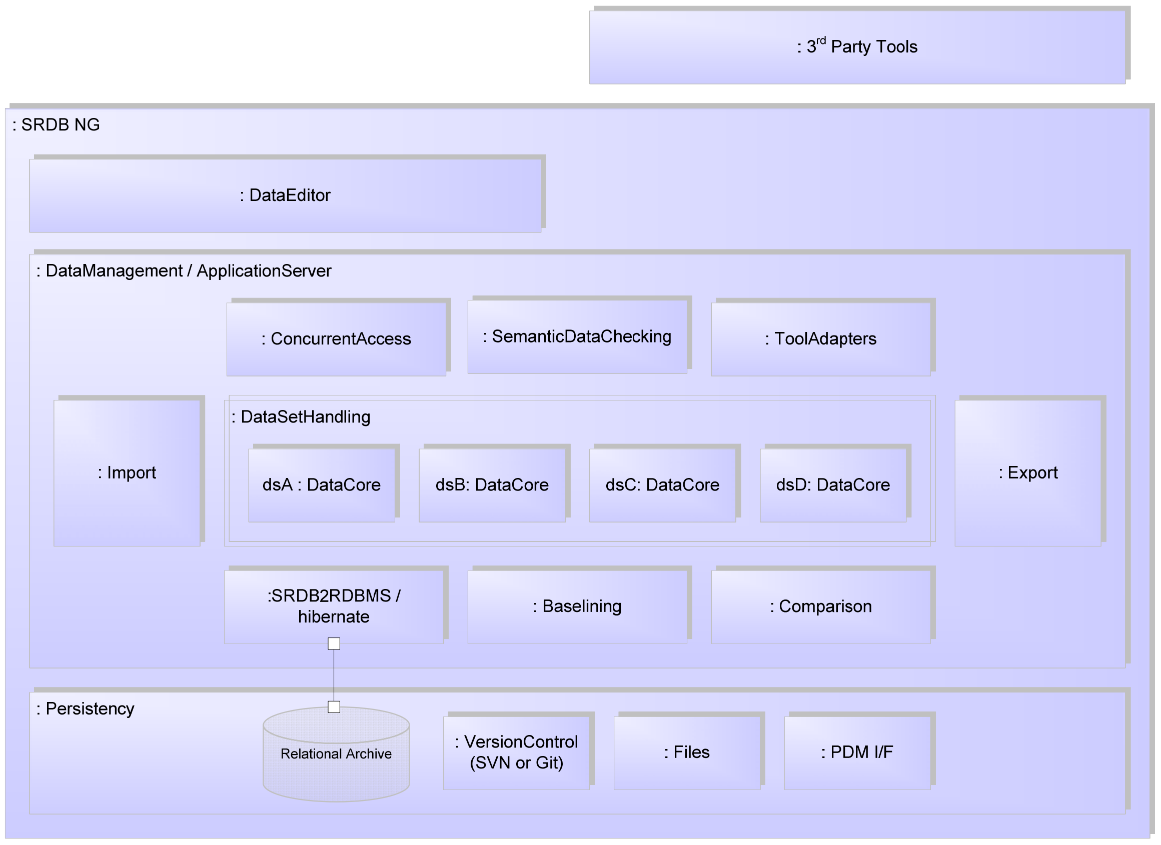

**Figure 5: SRDB NG High-level Architecture**

One of the problems in the past was to deal with different meta models, i.e. to realize import /export. It cannot be presumed that one single meta model is used along the customer supplier chain, for the internal data management representation and also along the customer supplier chain. Based on the experience gained on model transformation it is planned to apply a full model-2-model transformation, where each transformation layer is fully specified by a dedicated meta model. This allows then even for IO data in the source respectively the target data the functionality on checking or comparing.

Figure 6 gives an example for an import/export of data in the MIB format. The SRDB NG will have its own conceptual data model represented e.g. in UML (and thus independently from Ecore). An Ecore representation can be automatically derived by a model to model transformation,, from there the SRDB internal emf data core (consisting of emf.model and emf.edit) can be automatically derived. For a given IO format to be supported (in this case MIB), the approach would be exactly the same. A dedicated MIB conceptual data model would be developed also e.g. in UML. From there the MIB Ecore model could be derived and from there the mib.emf.datacore.

As the same transformators for both models are used, the I/F's – on a pattern level – for srdb.emf.datacore and mib.emf.datacore are exactly the same. This means that the framework for *.emf.validate can also be used, same for the compare. The io routines can be implemented as a plain data transformation in Java by using the emf.edit I/F.

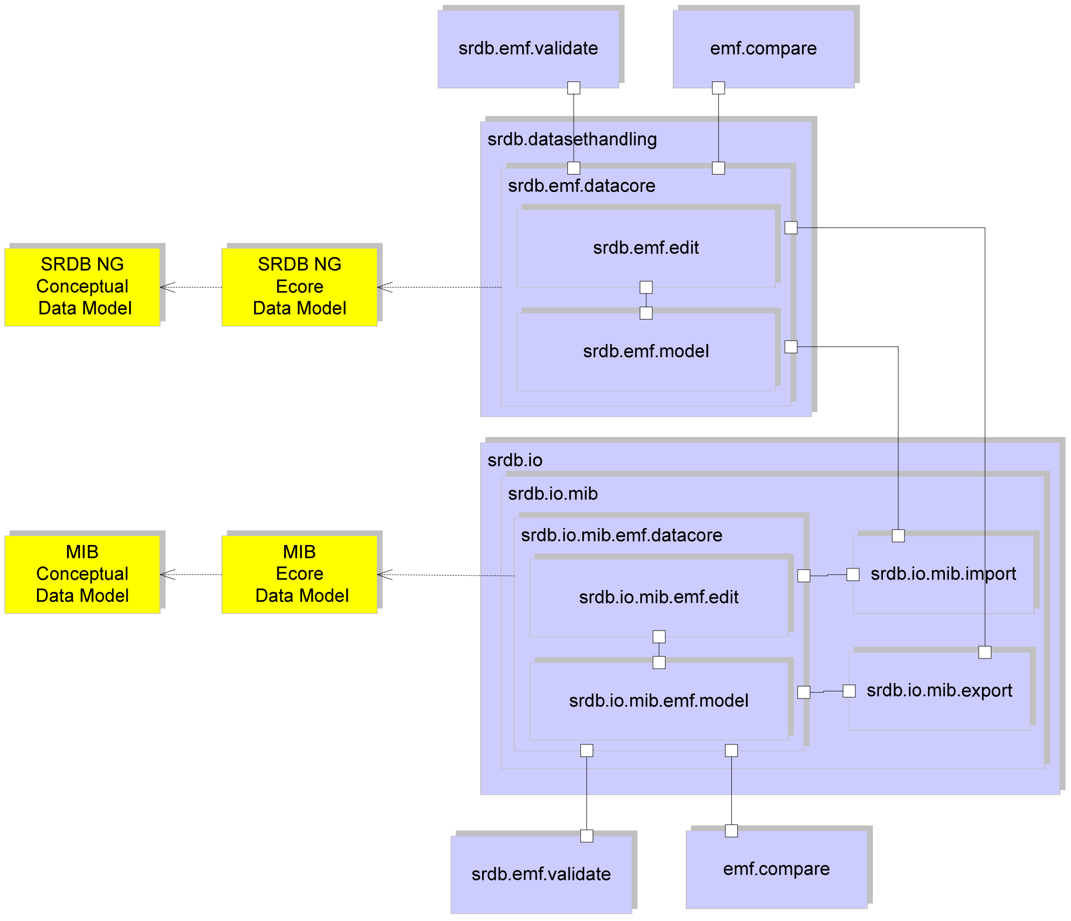

**Figure 6: Dealing with different meta models**

**ECSS-E-TM-10-23**

The ECSS E-TM-10-23 'engineering database' working group, was to specify (systems) engineering database in response to the related references in ECSS E-10. In the course of the definition it turned out that an industrial best practice cannot be taken and specified. Rather it turned out that still:

- Many very valuable information items are kept in document instead of models
- An tool and implementation technology independent specification of data is not in use
- Existing and use systems are in use for years and typically very cost intensive for development, maintenance and integration

Therefore the working group decided to identify and specify high-level concepts in order to increase the awareness, facilitate the discussion in order to overcome the current situation. One of the main deliverables is a conceptual data model developed in UML, to give a concrete example. The model meanwhile has been used in different activities in R&D studies but also in operational implementations.

While the working group commonly agreed on the concepts and the need for a conceptual data model, the group failed in a recommendation on the methodology or tools used for the definition of the conceptual data model. The approach taken for the E-TM-10-23 model is UML based on a simple UML profile. In different activities the approach and the model has been fully validated and proven. Nevertheless it turned out that the following short-comings should be taken into account: :

- Diagram orientation of UML turned out not necessarily being helpful in any case for the data model definition. Interesting other approaches could be Express or OWL.
- Model-management in particular versioning or comparison of models is difficult
- Meta model and model instances i.e. model libraries cannot be kept and maintained in the same tool. This in particular applies for a full collection of required properties.
- Semantic expressiveness for data modelling is limited, but can be improved through profiling. However profiling typically doesn't increase the usability of a UML tool

For future enhancements it is important, not only to limit the modelling to data, but also take into account modelling of processes, organisation and views.

Considering the current preparation and planning of the different efforts a standardization of THE European Space Industry approach for data modelling would be more then helpful to reduce further costs in tool or methodology incompatibilities.

## VALIDATIONS PERFORMED

Over the last couple of years different R&D activities have been initiated and sponsored by ESA / ESTEC in the area of system database, in parallel to the ECSS E-10-23 standardization effort. Those activities covering different aspects clearly validated the different concepts and approaches identified in this paper. Among others those are:

- Space Systems Reference Model (SSRM) where the ECSS-E-TM-10-23 model has been used the first time and demonstrated:
  - Automated (MDA) generation of the conceptual data model into different database technologies
  - Service oriented architecture concepts for the deployment architecture
  - Exchange of TM/TC data (directly taken from ATV) through the developed database
  - Exchange of key mechanical properties through the developed database
- Validation support of E-TM-10-23: Validation of the product structure and operational design model through sample implementation in Eclipse EMF/GMF to demonstrate the conversion of existing diagram views into a formalized design view
- Virtual Spacecraft Design (VSD): Objective of VSD is to demonstrate the improvements of system engineering processes by the introduction of a virtual product model into the design process. The validated elements include:
  - Validation of the architectural concepts for data management functions implemented by using Eclipse resources
  - QUDV model
  - Increased validation of the E-TM-10-23 data model by a thorough document review of the sample project in the preparation of the conceptual data model
  - Using a reference architecture for the different VSD building blocks with shared elements in particular having a common data core

## CONCLUSION

The SRDB is a key element supporting the overall S/C design, integration, verification and operation. The current tools are meanwhile matured to an extend, where an economic customization and use seems not to be granted. Therefore it is a top priority for Astrium to supersede the existing system. Based on in particular the joint activities with ESA in frame for TRP and ECSS activities, the relevant concepts, methodologies and technologies have been identified and demonstrated. An initial validation is also included in that. However some more validation activities need to be performed as in particular TM/TC data has not been addressed so far.

The Astrium activities for the SRDB already have been started. In 2010 the focus is on the user requirements definition and initial architecture on technology considerations. The K/O for the developments is planned beginning of 2011. The main areas covered in 2011 are design, continued validation and in particular conceptual data model definition. For 2012 it is planned to proceed with the developments to a maturity and readiness, allowing a start of the pilot project with the SRDB NG in 2013.

The continuation of the ESA collaboration in the frame of the preparation and development is highly appreciated in order to prepare for the programmes in the best way. One of the very crucial items can only be solved in a close coordination is the selection of the data modelling methodology and in particular the formalization of the current data models for TM/TC data as expressed in ECSS E-70-31, XTCE, ECSS E-70-41 or MIB.