# Formal Modeling of Data.
# A Case Study for Space Applications[1]

Jean-Paul Blanquart[1], Gérard Bulsa[2], David Lesens[3], George Mamais[4], Maxime Perrotin[2]

[1] Astrium Satellites. 31, avenue des cosmonautes, F-31 402 Toulouse Cedex 4, France
Jean-Paul.Blanquart@astrium.eads.net
[2] ESTEC. Keplerlaan 1. PO Box 299, NL-2200 AG Noordwijk, The Netherlands
{Gerard.Bulsa, Maxime.Perrotin}@esa.int
[3] Astrium Space Transportation. Route de Verneuil, BP 3002, F-78 133 Les Mureaux Cedex, France
David.Lesens@astrium.eads.net
[4] Semantix Information Technologies, K.Tsaldari 62, Poligono 114 76, Athens, Greece
gmamais@semantix.gr

**Abstract**. This paper reports on a case study investigating the interest of formal data modeling approaches for on-board space software. The objectives are to complement formal approaches on functional parts and solve difficulties coming from lack of formalism in the description and handling of data exchanged between different parts of software. The first part addresses the analysis of space software data and their classification into families (section 2) and the elaboration of an ASN.1 data model for a case study representative of these data families (section 3). The second part illustrates two important outcomes of such formal data models: the capability to automatically generate "Interface Control Documents", the contractual documents describing software interfaces and data (section 4) and automatically generate the needed interfacing code with the appropriate format encoders and decoders (section 5).

## 1 Introduction

The use of modeling techniques for describing on-board and ground space systems has been identified by many studies as an appropriate way to master the complexity of software. Today, projects show an increasing interest for applying a model-based approach, from which several benefits are expected:

- A clear and unambiguous representation of the software architecture;
- A complete and verifiable representation of the software behavior;
- A reliable implementation of the software.

There exist several languages and tools that have been assessed as good candidates to fulfill most of these needs. These languages allow the description of a system's logical and physical architecture as well as its behavior, and tools provide simulation, model checking, automatic test and code generation.

However, a system is generally not developed in an independent manner by a single developer team. Whether they use modeling techniques or not, many projects encounter

[1] This work has been partially funded by the ASSERT project (Contract n°: 004033 of FP6-2003-IST-2) [ASSERT] and the ESA project SSCDMT "System Software Co-engineering: Data Modeling Technologies" (Contract ESTEC 20467/06/NL/JD) [SSCDMT].