

# Trustworthy aerospace systems

*The challenge to achieve an integrated, coherent approach...*

**B**uilding modern aerospace systems is highly demanding. They should be extremely dependable. They must offer service without interruption (ie. without failure) for a very long time – typically years or decades. Whereas ‘five nines’ dependability, ie. a 99.999% availability, is satisfactory for most safety-critical systems, for on-board systems it is not. Faults are costly and may severely damage reputations. Dramatic examples are known. Fatal defects in the control software of the Ariane 5 rocket and the Mars Pathfinder have led to headlines in newspapers all over the world. Rigorous design support and analysis techniques are called for. Bugs must be found as early as possible in the design process while performance and reliability guarantees need to be checked whenever possible. The effect of Fault Diagnosis, Isolation and Recovery (FDIR) measures must be quantifiable.

Tailored effective techniques exist for specific system-level aspects. Peer reviewing and extensive testing find most of the software bugs, performance is checked using queuing networks or simulation, and hardware safety levels are analysed using a profiled Failure Modes and Effects Analysis (FMEA) approach. Fine. But how is the consistency of the analysis results ensured? What is the relevance of a zero-bug confirmation if its analysis is based on a system view that ignores critical performance bottlenecks? There is a clear need for an integrated, coherent approach.

This is easier said than done. The inherently heterogeneous character of on-board systems involving software, sensors, actuators, hydraulics, electrical components, etc., each



© 2010 ESA-CNES-ARIANESPACE/Photo Optique Vidéo CSG

with its own specific development approach, severely complicates this.

## Modelling

About three years ago we took up this grand challenge. Within the ESA-funded COMPASS (COorrectness, Modelling and Performance of Aerospace SyStems) project, an overarching model-based approach has been developed. The key is to model on-board systems at an adequate level of abstraction using a general-purpose modelling and specification formalism based on Architecture Analysis & Design

Language (AADL) as standardised by SAE International. This enables engineers to use an industry-standard, textual and graphical notation with precise semantics to model system designs, including hardware as well as software components. Ambiguities about the meaning of designs are abandoned.

Among the system aspects that can be modelled are (timed) hardware operations, specified on the level of processors, buses, etc.; software operations, supporting concepts such as processes and threads; hybrid aspects, ie. continuous, real-valued variables with (linear) time-dependent



dynamics; and faults with probabilistic failure rates and their propagation between components. A complete system specification describes three parts: nominal behaviour, error behaviour, and a fault injection specifying how the error behaviour influences the system's nominal behaviour. Systems are described in a component-based manner such that the structure of system models strongly resembles the real system's structure.

### Analysis

This coherent and multidisciplinary modelling approach is complemented by a comprehensive palette of analysis techniques. The richness of the AADL dialect gives the power to specify and generate a single system model that can be analysed for multiple qualities: reliability, availability, safety, performance, and their mixture. All analysis outcomes are related to the same system's perspective, thus ensuring compatibility. First and foremost, mathematical techniques are used to enable an early integration of bug hunting in the design process. This reduces the time that is typically spent on a posteriori testing – in on-board systems, more time and effort is spent on verification than on construction – and allows for early adaptations of the design. The true power of the applied techniques is their almost full automation: once a model and a property (eg. can a system ever reach a state in which the system cannot progress?) are given, running the analysis is push-button technology. In case the property is violated, diagnostic feedback is provided in

terms of a counterexample, which is helpful to find the cause of the property refutation. These techniques are based on a full state space exploration, and detect all kinds of bugs, in particular also those that are due to the intricacies of concurrency: multiple threads acting on shared data structures. These types of bugs are becoming increasingly frequent, as multithreading grows at a staggering rate.

Whereas academic tools rely on properties defined in mathematical logic, a language that is major obstacle for usage by design engineers, COMPASS uses specification patterns. These patterns act as parameterised 'templates' to the engineers and thus offer a comprehensible and easy-to-use framework for requirement specification. In order to ensure the quality of requirements, they can be validated independently of the system model. This includes property consistency (ie. checking that requirements do not exclude each other), and property assertion (ie. checking whether an assertion is a logical consequence of the requirements).

Assessing system safety and dependability requires the analysis of potential failure modes within a system. This is supported by both top-down techniques such as (dynamic) Fault Tree Analysis (FTA), and bottom-up methods such as (dynamic) Failure Modes and Effects Analysis (FMEA). Both are completed by stochastic analyses that allow evaluation of the fault tolerance of the given system, and estimation of the criticality of failure modes by taking the severity of their consequences into account. System models can include a formal description of both the fault detection and isolation of subsystems, and the recovery actions to be taken. Based on these models, analysis capabilities are provided to evaluate the operational effectiveness of the FDIR measures, and to assess whether it is sufficient to observe the

system parameters in order to make failure situations diagnosable.

All techniques and the full modelling approach are supported by the COMPASS toolset, developed in cooperation with the Italian research institute Fondazione Bruno Kessler in Trento, and freely downloadable for all ESA countries from the website mentioned at the end of this text.

### Evaluation

The COMPASS approach and toolset was intensively tested on serious industrial cases by Thales Alenia Space in Cannes (France). These cases include thermal regulation in satellites and satellite mode management with its associated FDIR strategy. It was concluded that the modelling approach based on AADL provides sufficient expressiveness to model all hardware and software subsystems in satellite avionics. The hierarchical structure of specifications and the component-based paradigm enables the reuse of models. Also incremental modelling is very well supported. The reliability, availability and safety analyses as provided by the toolset were found to be mature enough to be adopted by industry, and its results allowed for the evaluation of design alternatives. Current investigations indicate that the integrated COMPASS approach significantly reduces the time and cost for safety analysis compared to traditional on-board design processes.

Professor Dr Ir Joost-Pieter Katoen  
katoen@cs.rwth-aachen.de

PD Dr Thomas Noll  
noll@cs.rwth-aachen.de

Software Modelling and Verification  
Department of Computer Science  
RWTH Aachen University  
52056 Aachen  
Germany

Tel: +49 241 80 21200

compass.informatik.rwth-aachen.de