Successful Application of New Technologies to the Rosetta and Mars Express Simulators

J.E. Martin, R. Lowe, J.J. Osborne, A. Young, S. Thorn and P.Matakidis

VEGA Informations-Technologien-GmbH, Hilpertstr. 20A, D-64295 Darmstadt

jim.martin@vega.de

Abstract

VEGA IT GmbH is currently in the final stages of implementing the Rosetta and Mars Express simulators for the European Space Operation Centre (ESOC). Both simulators are high fidelity, entirely software-based tools principally used for training the operations team. This paper describes aspects of the development of the simulator, in particular focussing upon the application of new technologies, the re-use of previously developed models and infrastructures, and the potential re-use of the new technologies in future simulators.

1 Introduction

VEGA IT GmbH won the contract to supply both the Rosetta and Mars Express simulators to ESOC. The two simulators were to be developed concurrently and with a fixed schedule. Due to hardware re-use on the spacecraft, VEGA was able to establish a common code-base for the two simulators, which has simplified the overall development.

Each simulator contains a high fidelity model of the spacecraft platform. In comparison, the scientific payload instruments have been modelled in a generic, lower fidelity manner, but incorporate a higher degree of user configurability. Such an approach allows the payload models to be readily updated should the need arise.

In response to customer requirements, and as a result of simulator design, it was found necessary to utilize a host of new technologies, not least the infrastructure within which the simulators run. However, all parts of the simulators are not new. In fact, the design also incorporates models and infrastructures re-used from other VEGA and ESA projects.

Although the use of SIMSAT-NT as a key infrastructure component was mandated, it was not available until 3 months after the beginning of the project (kick-off was October 1999). Initially, the simulator was targeted at a purely Alpha platform. Unfortunately, Compaq withdrew support for Windows-NT on an Alpha platform during the first year of simulator development. This forced a move to Intel based technology. Later in 2000, SIMSAT-NT was delivered for Intel and a new architecture was adopted for the simulator. From this point onwards, the spacecraft processor emulators would be run on an Alpha co-processor card mounted inside an Intel host machine. Spacecraft Model development continued throughout 2001 and both simulators are now being used to support mission preparations.

This paper aims to highlight the beneficial impact and development problems presented by the most interesting of these new and re-used technologies and infrastructures. Where new methodologies and technologies have been developed, the potential for their re-use is also presented.

2 Requested Infrastructure

Three technologies were stipulated as mandatory in the development of the Rosetta and Mars Express simulators. These were:

- Use of SIMSAT NT as an infrastructure.
- Use of 1750 processor emulation, enabling the execution of the real spacecraft flight software in the simulator.
- Conformance to the Simulation Model Portability Standard (SMP), allowing execution of the simulator within any SMP compliant infrastructure (such as SIMSAT NT or EuroSim).

It was a customer mandate that both the Rosetta and the Mars Express simulators be developed for an entirely new simulations infrastructure, SIMSAT-NT. SIMSAT-NT, developed by VEGA IT GmbH, is an infrastructure for running real-time simulations. Based on the Microsoft Component Object Model (COM), it runs under the Windows NT operating system and provides a number of features to assist simulation development.

The Rosetta and Mars Express simulators are the first ESA simulators to be developed using SIMSAT-NT. Therefore, the use of this infrastructure is the first example of the application of a new technology in this programme.

At the heart of SIMSAT-NT is a run-time Kernel, which provides simulation services such as a scheduler to execute simulator models in real-time and perform associated timekeeping activities. Both cyclic and discrete events are handled and faster than real-time execution is supported (depending upon workstation performance). A logger allows simulator event information to be recorded in a log and the simulator state vector can be stored and restored. SIMSAT-NT supports ActiveX scripting languages (e.g. JavaScript and VBScript) for controlling the simulation.

Full support of the Simulation Model Portability (SMP) standard is provided through the use of SMI. The advantage of this is to allow reuse of SMI compliant models e.g. Ground-NT and PEM-NT from previous generations of simulators. In addition, the Rosetta and Mars Express simulators are inherently SMI compliant, allowing them to be run using any SMI compliant infrastructure, e.g. EuroSim.

A Windows based Man Machine Interface (MMI) provides a user-friendly interface to the simulation allowing control of the simulation and data visualisation. A tree view allows easy browsing of simulator objects, and aliases can be created to permit easy access to these objects and their data. Alphanumeric displays (AND) and graphs assist the visualization of these data.

A major challenge of using SIMSAT-NT was the use of a graphical modelling tool (CAE - ROSE), which produces non-SMI compliant code. The method used to overcome this issue is discussed in section 3.
The use of 1750 Emulator technology is discussed later in section 4.

3 Simulator Architecture

The Rosetta and Mars Express simulators use an identical architecture, allowing a great deal of code and design re-use between the two projects. The simulator comprises the simulation models, i.e. those models that imitate the actual spacecraft sub-systems, the environment (orbits, planetary data, etc.), and the ground equipment. All links to the SIMSAT-NT MMI and Kernel are through the SMI. In comparison, communication between individual models may either be through direct interfaces such as function invocations, or through the SMI layer. User interactions, for example user commands, come through the MMI, into the SMI and thence into the target model. Figure 1 illustrates the architecture of the simulator.

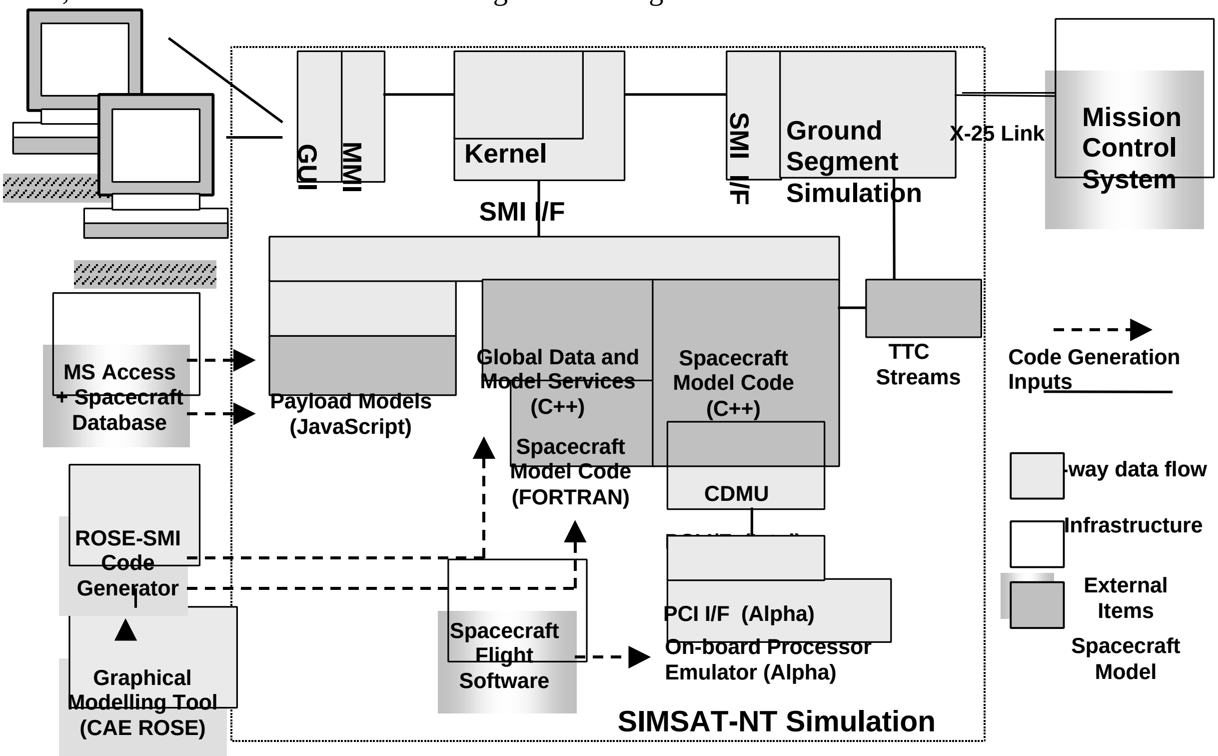

Figure 1: Architecture Over-view for Rosetta and Mars Express Spacecraft Simulators

The architecture of the Rosetta and Mars Express simulators are identical. Primary development work is carried out for Rosetta and then specific modelling changes included for Mars Express. This approach has allowed significant cost and time savings for the Mars Express Simulator development.

The simulation models can be grouped in terms of modelling technique. There are three major components: the spacecraft models, the on-board processor emulator and the payload models. A brief description of each follows.

3.1 Spacecraft Models

All the main spacecraft platform models have been developed using C++ and CAE ROSE. The graphical modelling tool has been used as a means of reducing model development time and promoting model design visibility during the development and maintenance phases. As mentioned in section 2, the use of a graphical modelling tool presented some challenges. In order to use the models in SIMSAT-NT it was necessary to develop a process to ensure SMI compliance. This challenge was overcome by developing a utility, written in *lex* and *yacc*, to parse the auto-generated code, extract the model objects, their data and services and wrap them in an SMI compliant layer. This process provides a well-established method that could be simply extended to use any code-generating modelling tool within SIMSAT-NT, e.g. MATLAB. Figure 2 shows an overview of the architecture developed for integrating the graphically developed models into the main spacecraft model.

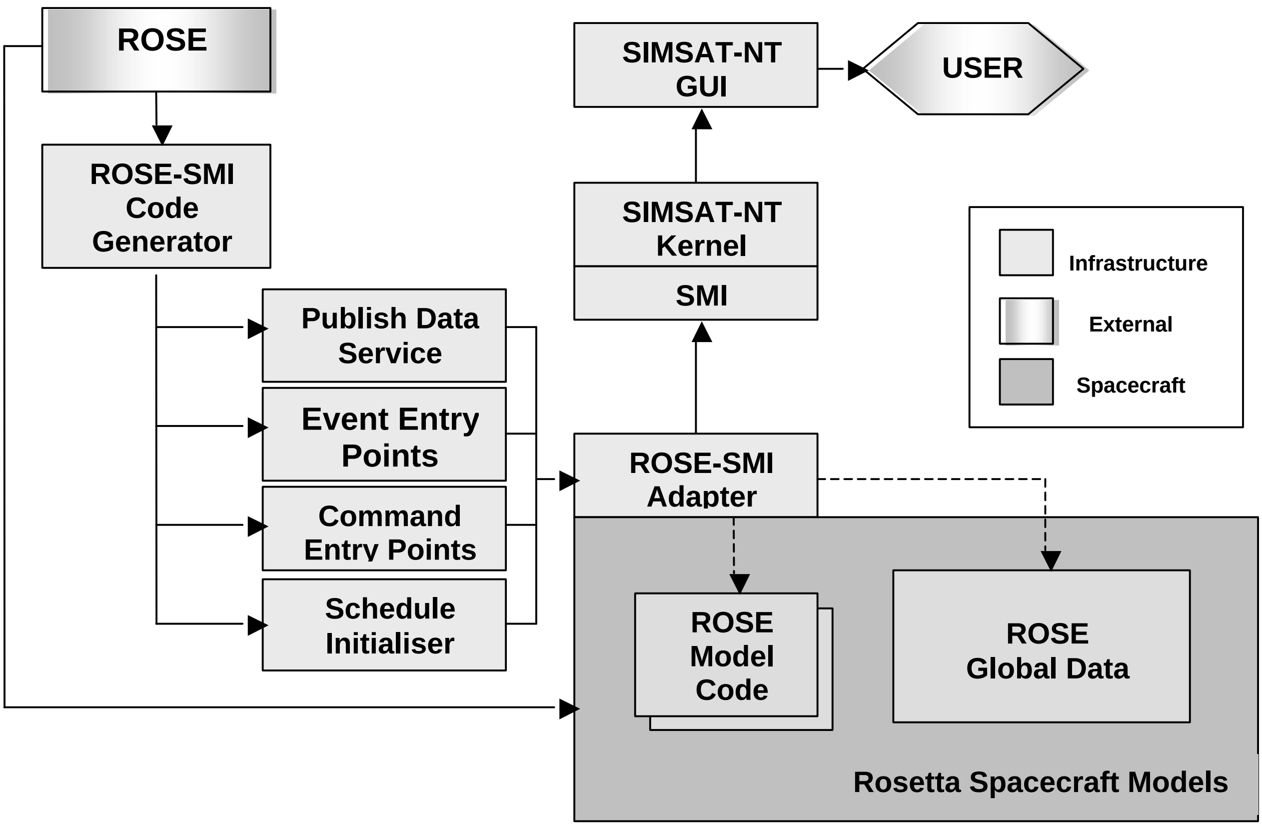

Figure 2: Generation of SMI compliant code from a non-compliant Graphical Modelling Tool
FORTRAN code generated by the Modelling tool (CAE ROSE) is parsed using software developed in *lex* and *yacc* and important services and data items are published to the SMI. Cyclic and one-off events can be scheduled in the simulator via the ROSE-SMI adapter.

3.2 Emulated On-Board Processors

The simulator contains an emulation of the processing modules of the spacecraft. The emulators are written in a combination of Ada and Alpha assembler. It was necessary to overcome several problems to utilize the emulators on an Intel platform, because they were designed to run on an Alpha processor. The emulator technology is described in more detail in section 4.

3.3 Generic Payload Models

The payload models have been developed by customising generic models produced directly from the spacecraft database (in XML and JavaScript). Customisation of each model is performed using JavaScript routines. Payload Modelling is described in more detail in section 5.

3.4 Progression of Technologies – Past, Present and Future

The table below shows the progression of technologies used in Spacecraft simulators in recent years. The development of the Rosetta/Mars Express simulators has taken several major steps forward from previous generations. In doing so, the project has 'broken ground' for the next generation. Development work on a simulator for ESA's Cryosat mission is already under way at Vega IT Gmbh. The Cryosat simulator is being developed using many of the new technologies tested and developed within the Rosetta/Mars Express simulator project.

Period	**-> 1999**	**1999-2002**	**2002 ->**
Simulator:	**ARTEMIS ENVISAT XMM INTEGRAL**	**Rosetta Mars Express**	**Cryosat**
Infrastructure:	SIMSAT VMS	SIMSAT-NT	SIMSAT-2000
Operating System:	OpenVMS	Windows-NT	Windows-2000
Emulation:	1750 Software Emulator	1750 Software Emulator	ERC32 Software Emulator
Platform:	Alpha Workstation	Intel with Alpha Co-processor	Intel
Programming Languages and methods:	Ada Macro-64	C/C++ SMI FORTRAN EMMA XML JavaScript	C/C++ SMI XML JavaScript

Table 1: Progression of Simulation Technologies is recent years

Taken as a whole, the majority of the new and re-used technologies in the Rosetta and Mars Express simulators lie in the particular application of the processor emulators and in the generic modelling applied to the payloads.

In addition the simulator also includes two ESA developed, SMI compliant models. The Position and Environment Model (PEM-NT) is a generic model for calculating interplanetary trajectories, and is designed to support future missions. Similarly, the ground models (Ground–NT) were developed to enable realistic connections between the simulated spacecraft and the Mission Control System (MCS) at ESOC.

4 Emulation in simulation

Rosetta has a complicated, multiply redundant data handling system, at the heart of which are four MA1750B standard processor modules. Up to two of these processor modules may be operational at any one time, one running the data management software, the other the attitude control software. It was mandated by the customer that emulation of the processor models be provided. Thus, the simulator contains two instances of a tailorable 1750 emulator developed previously by VEGA IT GmbH under contract for ESOC/ESA.

Using emulators has many benefits over functional and state models, and the usage of flight hardware. Firstly, the actual on-board software (OBSW) can be loaded and executed in real time. This means that the operation of the data-handling subsystem is as realistic as feasible, and allows the simulator to perform all the operations supported by the actual hardware. Secondly, it allows the simulation to be used to test new software patches before they are up-loaded onto the real spacecraft. This takes the simulator from being a pure training tool, to one able to support all operations throughout the life of the mission. Thirdly, the emulator allows for faults to be created, Input-Outputs to be invented and trapped, and the processor memory to be forced. Such a capability allows for better testing of the operations team and their procedures. Lastly, performance permitting, the emulator may be run faster than real time to allow accelerated testing.

The target platform of both simulators was Intel. However, the 1750 emulators were developed to run on an Alpha processor. Several methods were examined for establishing communications between the two processes (emulators and host) running under two physically separate, and entirely different processors. Solutions included hosting the emulators on an Alpha platform and using TCP/IP to connect to the rest of the simulator; and using a dedicated Alpha co-processor card within the Intel machine communicating via the PCI bus. Withdrawal of Microsoft support for its NT operating system on Alpha forced the decision to adopt the co-processor card option.

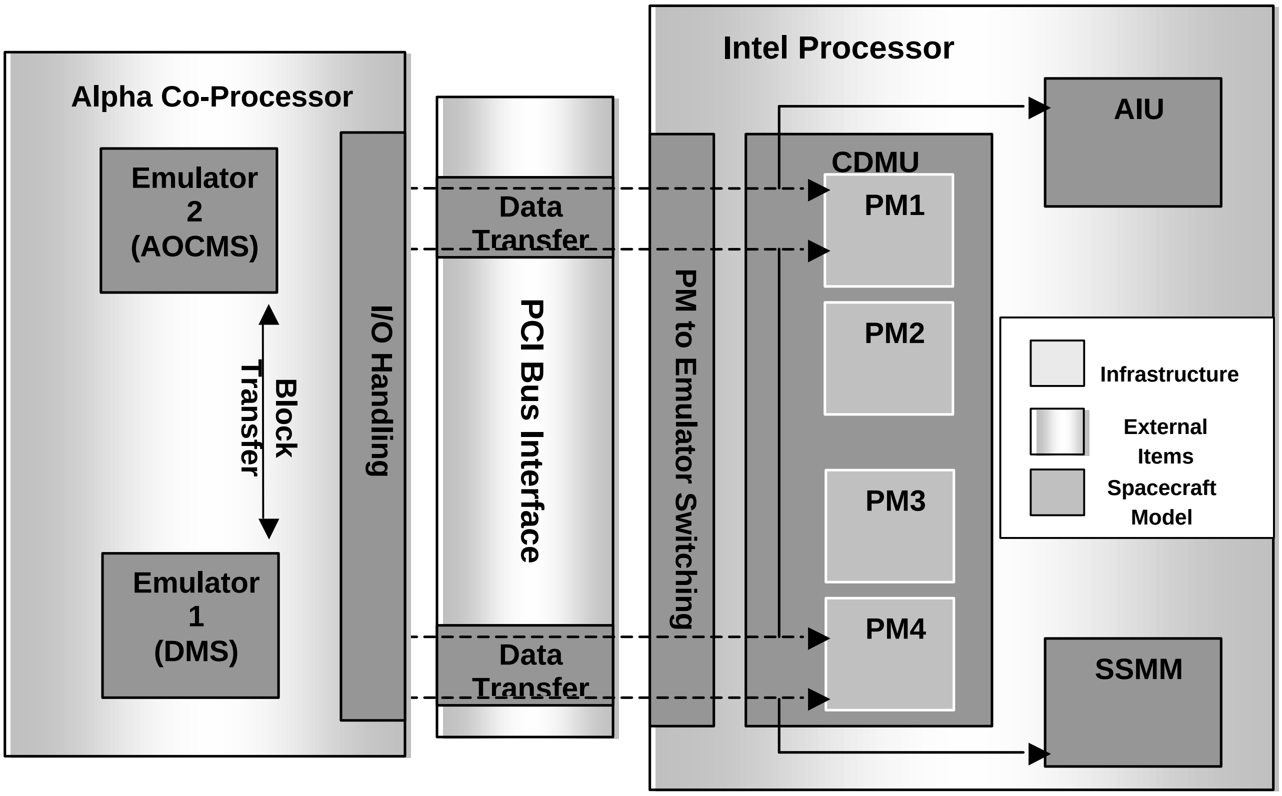

Figure 3: Architecture of Emulator to Spacecraft Model Interface

The 1750 Emulator used in the Rosetta and Mars Express simulators was developed to run on an Alpha processor. An Alpha co-processor card is interfaced to the main spacecraft simulation (on the host Intel machine) via PCI bus.

Figure 3 shows how the emulators are integrated into the simulator. For a broader perspective, see figure 1, section 3. Models of the Central Data Management Unit (CDMU) hardware are executed on the host Intel machine. Emulators for AOCMS and DMS flight-software are run on an Alpha co-processor card. The co-processor card is slaved to the host allowing synchronisation of the On-Board Software with the rest of the simulation.

During normal operations, communication between the DMS and AOCMS is handled across the PCI bus, via the Processor Module (PM) models. In cases where large blocks of data must be transferred between the two emulators, the data is passed purely within the co-processor card. This allows greater rates of data transfer with less risk of exceeding the PCI bus' data rate. Notification of the transfer is then passed to the Processor models on the host machine.

5 Generic Payload Modelling

A generic modelling approach has been adopted for the Rosetta and Mars Express (ROSMEX) scientific instruments, such that these models are based upon a generic satellite instrument, which allows for user re-configuration and logic changes. In total nineteen instruments are being developed using this technique, eleven for the Rosetta and eight for the Mars Express simulators. The basis of all instrument models is the re-use of the AQUA Infrastructure, essentially a telemetry/telecommand generator, which was developed by VEGA IT GmbH for the payload simulator of NASA's AQUA mission. This infrastructure provides a generic model in which telecommands (TCs) and telemetries (TMs) are specified in XML format.

Within the ROSMEX project, generation of the instrument model files was assisted by the development (in Microsoft Access) of a code generator. The code output by the generator is a template instrument model in an already executable state, capable of responding affirmatively to all TCs. The effort of the development task is thus concentrated on adding functionality to the already existing model. Additionally, as newer versions of the spacecraft database are received, the turnkey code generation phase is repeated to provide the latest XML, the upper level of TC handler functionality and JavaScript templates for any new TCs and TMs.

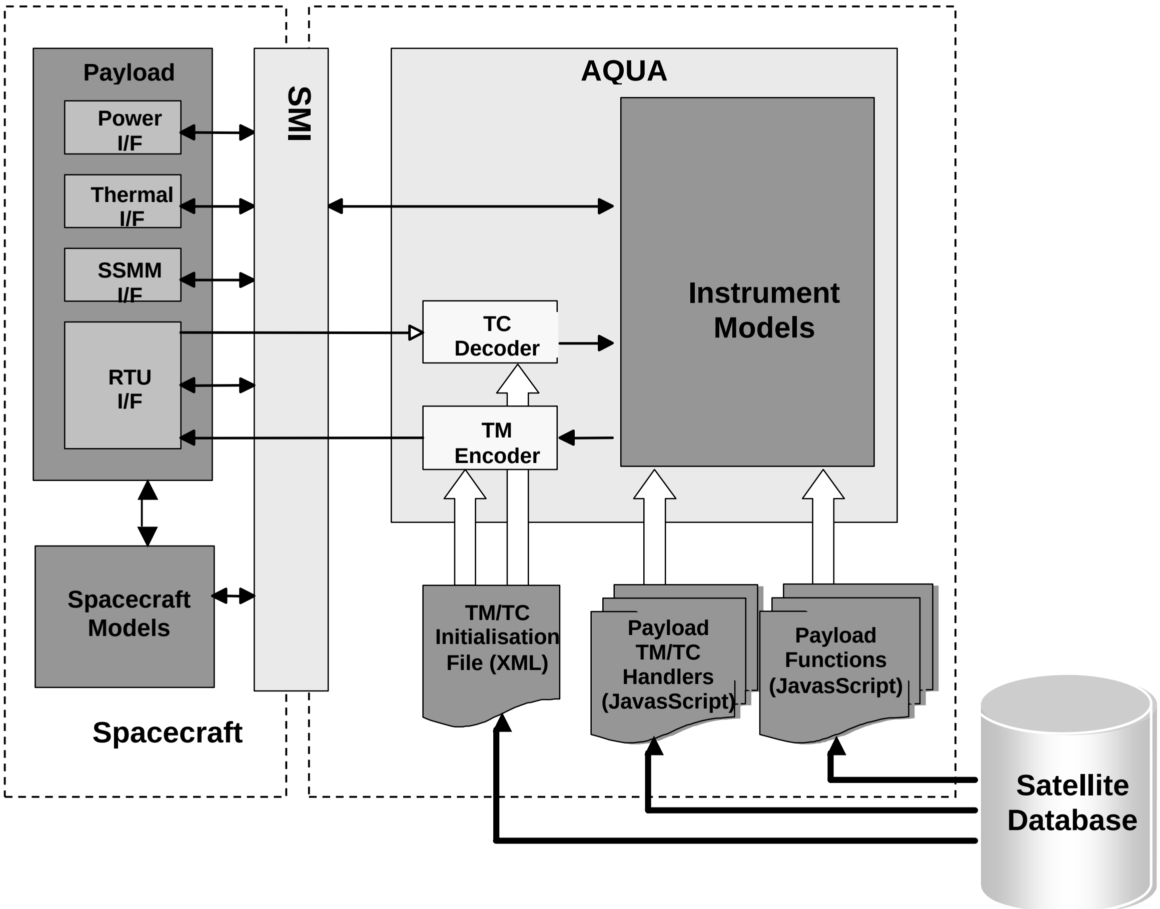

Figure 4: Payload Model Generation and Interfacing

A Payload Generator Database produces XML and JavaScript output, describing all TM and TC handled by a particular payload. Another layer of JavaScript is then hand-written to describe the operational behaviour of the instrument. The complete Payload model is then interfaced to the rest of the simulator via both SMI and direct model-to-model links.

A significant advantage of the Aqua Infrastructure and SIMSAT-NT is the ability to use mark-up and scripting languages, thereby removing the need to re-compile an instrument model should it be modified.

Benefits of this generic development approach include lower initial implementation costs and lower through-life maintenance costs. These cost savings are derived through a combination of reduced complexity of the JavaScript implementation (for example in comparison to a typical C++ implementation) and the opportunity for updating the functional behaviour without a new delivery of the Simulator. Furthermore, the models benefit from a commonality of concept, ease of implementation, rapid development cycle, intuitiveness to user and developer alike, and the potential for extended re-use beyond the current implementation.

6 Conclusions

This paper has highlighted the successful use of many new technologies in the Rosetta and Mars Express simulator programmes.

The infrastructure used to support the simulators, SIMSAT-NT, was itself the first new technology applied in the projects. As a result of the validation of SIMSAT-NT, it is currently being applied to new satellite simulators (CRYOSAT and RADARSAT) and has been taken outside of the space-domain as part of ESA’s technology transfer scheme into the chemical industry.

Application of processor emulation within simulation is itself not new, but the technique of using a co-processor card specifically for the emulators most certainly is. Whilst this technology has the scope for application to new projects, VEGA is also currently completing the porting of the emulators to the Intel platform, enabling new projects to take full advantage of the increases in the speed of these processors seen in recent times.

The payload models in the ROSMEX project re-used the Aqua Infrastructure developed for NASA, but also extended it beyond that baseline. As a result of the winning combination of XML definition and JavaScript functionality, the methodology developed by the ROSMEX team is to be adopted for the CRYOSAT simulator. However, instead of using the technique for payload models, CRYOSAT will use the infrastructure to rapidly develop a model of the data-handling system as a precursor to a full C++ model. In this way the customer may be given a TM/TC model, and provide feedback to VEGA, before the actual development has begun.

Much of what has been accomplished by VEGA on the Rosetta and Mars Express simulators was enabled by the SMI implementation of the SMP standard. The services offered by this interface have allowed many models, of many different architectures and implementations, to be easily integrated together into a working simulation.

7 Definition of Terms

1750	MIL-STD-1750 processor
AIU	AOCS Interface Unit
AND	Alpha-Numeric Display
AOCMS	Attitude and Orbit Control and Management Software
AQUA	TM/TC infrastructure packet (Named after the NASA Mission where is was first used for Payload modelling)
CDMU	Central Data Management Unit
COM	Component Object Model
DMS	Data Management Software
EMMA	Extensible Meta Macro Assembler (used for generating Alpha Assembler from a high level language)
ERC-32	Sparc processor
ESOC	European Space Operations Centre
Ground	Model of an ESA Ground Segment
GUI	Graphical User Interface
MCS	Mission Control System
MMI	Man Machine Interface
OBSW	Onboard Software
PEM	Position and Environment Model
PM	Processor Model
ROSE	Real-Time Object Oriented Simulations Environment (CAE Graphical Design Tool)
ROSMEX	ROSetta and Mars EXpress
SIMSAT	Simulations Infrastructure for Modelling SATellites
SMI	Simulation Model Interface (Implementation of the SMP)
SMP	Simulations Model Portability (ESA Standard)
SSMM	Solid State Mass Memory