The Swarm constellation simulator

A brand new, but still operationally responsive development

Torrance, CA
24th March 2009

Max Pignède [presenter], Mario Merri, Vemund Reggestad

European Space Agency (ESA) / European Space Operations Centre (ESOC)

Peter Fritzen, Michael Irvine

VEGA Deutschland GmbH & Co. KG / A Finmeccanica Company
What this presentation talks about...

1. Why attempting a new approach in making simulator software?
2. “Operationally responsive” architectures and technologies
   a. SMP2
   b. REFA
   c. EGOS-MF
3. Benefits for the users
4. Conclusions
1. Recognition that development strategies better than mere reuse should be explored
2. Take advantage of new technologies and of ESA most recent standards
3. ESA (ESOC) is preparing the first deployment of a new generation of operational simulators in the context of the Swarm mission, a constellation of three satellites
Overview of the Swarm mission

1. Study the dynamics of the Earth magnetic field and its temporal evolution

2. Improve understanding of its various contributing sources (Earth core, Earth mantle and lithosphere, magnetosphere and ionosphere)

3. Two low spacecraft (circa 450 km altitude, same inclination) separated in RAAN by 1.4° and in AN crossing by 10 seconds of each other

4. One higher spacecraft (530 km) in orbital plane of 0.6° inclination difference with respect to the lower pair

More details are available at: http://www.esa.int/esaLP/LPswarm.html
Why do we need a reference architecture?

Example of what a simulator software architecture may look like:

1. No clear interface between models
2. Difficult to isolate models for reuse
3. Tight coupling of database and models
4. Reuse potential mostly only via “copy&paste”

How about if we …

a. defined clear interfaces between the different elements in a spacecraft?

b. defined some suitable breakdown of a whole simulator into models?

c. improved reusability at model level in a “plug&play” fashion?
1. This is a computer running a satellite related application software
   a. it provides the operator with the same view on the satellite as of the real satellite
   b. what matters for ground operators is essentially TM and TC

2. A simulator includes typically:
   a. an on-board software emulator
   b. functional models of satellites hardware equipments
   c. and ... an operator interface

3. Typical simulator usage scenarios are ...
   a. mission control system and facilities testing
   b. validation of the operational procedures
   c. training of operations staff -- with e.g. injection of contingencies
“Operationally responsive” technologies

1. The SMP2 Standard ... 
   a. promotes portability of models among different simulation environments and operating systems 
   b. promotes the reuse of simulation models 
   c. fulfills these objectives by providing a standard interface between the simulation environment and the models 

2. The Reference Architecture (REFA) ... 
   a. identifies, using SMP2, a reference spacecraft simulator architecture which can be used as the basis for simulators design and development 
   b. achieves shorter development cycles by reuse relying extensively on a common architecture 
   c. fulfills these objectives by specifying interfaces between spacecraft subsystems and by identifying models which can be developed in a generic fashion
SMP2 architecture

SMP2 first component: a SIMULATION (with its model instances)

Model.1 <- Model.2 <- ... <- Model.i <- ... <- Model.n

Models interact with each other

Scheduler calls model entry points

Models call any simulation service

The interface offered by the model to the simulation environment does not change

SMP2 second component: a Simulation ENVIRONMENT (and its services)

Mandatory services:
- Information logging
- Entry point scheduling
- Times provision
- Events manager

Mandatory interfaces:
- Access to simulator state
- Access to simulator services
- Publication mechanism

The interface offered by the simulation environment to the model does not change
SMP2 simulation services and interfaces

1. Simulation services are...
   a. provided by the simulation environment
   b. consumed by the models
   c. based on standardised interfaces

2. Every model must implement the `IModel` interface

3. Every simulation service must implement the `IService` interface

```
IModel

ISimulator

IScheduler

ITimeKeeper

ILogger

IEventManager

Connect
Run
Hold
Restore
GetState
GetTimeKeeper
GetModel
...etc...

Connect
Configure
Publish
GetName
GetParent
...etc...
```

Environment

Service

Scheduler

Time Keeper

Logger

EventManager
SMP2 key benefits

1. SMP2 avoids developing models which use the operating system or hardware specific dependencies -- Platform Independent Model (PIM)

2. SMP2 promotes the use of modern software engineering techniques -- in particular component based design

3. SMP2 makes reuse easier via breaking dependencies between simulator models

4. SMP2 allows dynamic configuration -- for example the user may at “runtime”:
   a. select different orbit propagators depending on the required accuracy
   b. switch a component that simulates a hardware equipment with a component that interfaces with the real equipment (when moving from simulated models to real equipments)
SMP2 hierarchy -- the main parts

Catalogue C1
- Model M1
  - Fields ..
  - Operation ..
  - Entry Points ..
- Model M2
  - Field ..
  - Associations ..
  - Entry Points ..
- Model M3
  - Fields ..
  - Entry Point ..

Assembly A2
- Model-Instance C
  - Model M2

Assembly A1
- Model-Instance A
  - Model M1
  - Model M3
- Model-Instance B
  - Model M1
  - Model M4
  - Model M3

Schedule S1
- Assembly A3
- Assembly A1

Catalogues define SMP2 models library
Schedules define the scheduling of the entry points of model instances
Assemblies specify how model instances are integrated
1. REFA started with investigating what would be worth being standardised for all satellite subsystems across simulators
   -- via the actual screening of 4 different space missions
2. As a result REFA identified ...
   a. what should the “reference spacecraft simulator” requirements be
   b. all interfaces between the different elements in a spacecraft simulator
   c. the models which can be developed
      – in a generic fashion (e.g. satellite dynamics, orbital environment modelling, satellite thermal control, communications subsystem, ...)
      – those other models which need to be developed specifically for each mission
3. This is a system **architecture** -- some hard work remains to be done ...
4. Use this architecture as the basis for future simulators development!
REFA logical model

REFA subsystems:
- Payloads
- Electrical Power
- AOCS
- Data Handling

Radio Frequency
- Thermal Control
- Reaction Control
- Data Links

REFA utilities
- Simulation Monitor
- Parameter Mapper
- Configurator

Generic Units
- Parameters
- Events
- Failures
- Messages

Simulation Model Portability 2 (SMP2) Component Model
- Logger
- Scheduler
- Time Keeper
- Resolver
- Events Manager

Aligned Unit

Magnetic Torquer Head
REFA outputs -- example

The Magnetic Torquer Head requires an interface to read the Earth Magnetic Field.

The Magnetic Torquer Head requires an interface to provide an External Torque.

**Generic Models (SMP2)**
- **SIMDYN**
- **PEM**

**External Force Interface**

**Earth Magnetic Field Interface**
REFA outputs -- and SMP2 editor support
REFA for building a new simulator ...
1. Eclipse based IDE extension for developing SMP2 simulators and for increasing software development efficiency  
   a. EGOS-MF is a collection of Eclipse Plug-ins. It uses MagicDraw.

2. Supports the full life cycle of simulator development  
   a. design of models -- systematically (re-)start at the UML design stage!  
   b. code generation by SIMSAT MIE (including code merge)  
   c. document generation (with customisation of generation templates)  
   d. model execution and debugging

3. The answer for managing simulators complexity efficiently is ...  
   a. to model! -- and the model is the single source of information  
   b. to automate!  
   c. and not to repeat yourself!

4. Hence: Model Driven Software Development (MDSD) supported by the EGOS-MF suite
EGOS-MF overall development workflow

Requirements Engineering
- Requirements Editor
  - .csv
- Requirements Import Tool
- Catalogue Import Tool
  - .cat

Architecture and Interface Design
- UML Editor
  - UML Model
- Model Validation Tool
  - .cat

Interface Design
- Document Generation Tool
  - .doc
  - .html
- CORBA IDL Generation Tool
  - .idl
- XML Schema Generation Tool
  - .xsd
- Catalogue Generation Tool
  - .cat
  - Catalogue Validator (MIE)
- Catalogue Generator (MIE)
  - .cpp

Development
- doxygen
- C++ Compiler
- C++ Code Generator (MIE)
- UML Editor
- Architecture and Interface Design
EGOS-MF model driven generation simple case

### UML: Information Model

- **Structure**
  - `Packet`
    - `{order=0, Header}`
    - `{order=1, DataField}`

- **Sequence**
  - `Header`
    - `(itemType=Octet, size=6)`
  - `DataField`
    - `(itemType=Octet, maxSize=65536)`

### UML: Functional Model

- **Operation**
  - `IPacketDecode`
    - `Decode(packet : Packet)`
      - `Decoded(packet : Packet)`

- **Exception**
  - `DecodeFailed`
    - ` Reason : String8`

### XML Schema

### CORBA IDL
This systematic code generation process is illustrated next proceeding from the previous UML example

<<SMP2model>>MagneticTorquerHead
**EGOS-MF example: Catalogue Generation**

1. Catalogue Generator translated UML Design into SMP2 Catalogue (XML File)
2. Code Generator generates C++ Source Code for each SMP2 Model with SIMSAT MIE
3. Document Generator generates Design Documentation for each SMP2 Model

1.1.1.3 Function
This is the model of a magnetic torquer head.
It is in charge of computing the produced torque depending on the dipole momentum and on the magnetic field.
The produced torque is then set into the SIMDYN generic model.

1.1.1.4 Subordinates

1.1.1.5 Dependencies

<table>
<thead>
<tr>
<th>Reference</th>
<th>Interface</th>
<th>Multiplicity</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>EarthField</td>
<td>OESM:BEM:EEarthMagneticField</td>
<td></td>
<td>The interface to use the Earth field</td>
</tr>
</tbody>
</table>
1. Users have identical views on ...
   a. Message logging features
   b. Model failures
   c. Parameter limits
   d. Simulation tree search

2. Users have identical views (for example) on failed or forced parameters
   a. The magnetic torquer head has Analogue Parameters (e.g., the global magnetic field) -- hence they have limits and they can be forced
   b. The magnetic torquer head can be failed because it is a Generic Unit
Conclusions

1. Building the ESA operational simulators is “better, faster, cheaper” with:
   a. SMP2 -- interfaces enabling reuse and exchange of satellite models
   b. REFA -- structured reuse of past acquired simulators knowledge
   c. EGOS-MF -- automatised component based development
   d. SIMSAT -- solid software infrastructure on linux platforms

2. From the users perspective
   a. These advanced technologies respond to project needs and schedules
   b. Requirements for multi-satellite systems are satisfied
   c. Demonstrated potential for reduced costs in producing future simulators
Thank you for your attention!

Any questions...??..