Optronics Systems Specification
Mission and High-level Behavior co-validation

Jean-Charles Causse
Pascal Lebaillif
Jean-Marc Perey
Jean-François Le Roux
Xavier Warzee

Scope and Goals
THOMSON-CSF OPTRONIQUE

Scope

— New missions increase demands on system adaptability to environment constraints leading
  to support complex system-level functions such as multi-sensors data acquisitions, multi-bands data fusion.

— Digital processing and opto-mechanical sub-systems/components support more and more functions leading
  to an increase complexity to predict the global behavior and performances of the system before any development or integration.

Goals

— Use « synthetic environment » simulations
  to validate system missions with the customer,
  to avoid « gap » with the customer and help him to describe the operational requirements.

— Use « executable formalisms » simulations
  to verify soundness of system control logic,
  to explicit concurrent properties between system functions.

— Connect both types of simulations
  to co-validate missions & high-level behavior.
Optronics Systems and Simulations Architecture
Generic IRST Optronic System

- Class of passive military infrared sensor systems.
- Reliably detect and track targets.
- Components:
  - Infrared sensor, IRST computer, track files database, display hardware.
- Inputs:
  - A continuous sequence of infrared image frames of background clutter with or without target.

IRST Simulation Architecture

Tactic Model  Target Model

Environment Model:
- atmospheric model
- ground model
- thermal model
- materials model

Sensor Model

Pilot  Carrier Model

Control logic

Image processing  Feedback control

New Line Of Sight, Sensors parameters (Integration time, etc.)

Carrier motion, Pilot actions
Current practice in system analysis and design

---

External functional analysis
- Operational scenarios definition
- Functional decomposition by mode (SA/RT)
- Performance allocation

Internal functional analysis
- Functional decomposition using technical solutions
- Input / Output definition
- Performance induction

System Architecture definition
- Components definition
- Functions allocated to components
THOMSON-CSF Optronique

SA/RT Exemple 2/2 (supervisor)

Raised problems

- No validation
  - No check of the consistency of the system behavior
  - Partial use of operational scenarios (no check of the completeness of the system behavior)

- No continuity between functional analysis and system architecture definition
  - New diagrams (consistency?)

- Several models, simulation tools, configuration management.
Current practice in «Distributed Modeling & Simulation»

DoD/DMSO

Two recent Defense Science Board (DSB) Task Forces (Technical Advanced Distributed Simulation and Resiliency, references 1) and (C2) have recommended that architecture efforts to combine live, virtual, and constructive simulation be expanded.

In addition, recent special studies have confirmed the need for architectural activities to promote the interoperability of multiple distributed simulations to support either traditional or commercial applications (Report of DSB, reference 2).

Interoperability and reuse are limited because the Department of Defense lacks a common technical framework for simulation architectures.

There is now a consensus that DoD must establish such a framework to facilitate the interoperability of all types of modeling and simulation running on distributed C4I Systems, as well as to facilitate the reuse of M&S components.

Reference 1.

Reference 2.

From the Simulation Based Design (SBD) project
The proposed Methodology

The co-validation approach

One model, three Views

Provide one model, with several views adapted to each development stages:

- A behavioral view
  - to check consistency of the system behavior,

- An operational view
  - to check completeness of the system behavior,

- An architectural view
  - to refine the system behavior in terms of components and a reference architecture.
The Behavioral View

— Complex systems such as multi-sensors Optronics systems have to be reactive to events coming from their environment:
  ♦ For instance, sensors Line Of Sight may have to change according to scene updates, pilot commands, data fusion results, target recognition, etc.
  ♦ How these events must be handled? Which priorities must be followed?
  ♦ How to ensure the specifications don't forget a particular combinaison of happening events?

— Since [Harel 1986], the statechart approach has shown to be useful to model reactive systems.
  ♦ Relatives researches : Gajski/SpecCharts (University of Stanford), Esterel/SyncCharts (INRIA/CMA), Ptolemy/“Charts (UC Berkeley).
  ♦ Hierarchical Finite State Machines (FSMs) to process events,
  ♦ concurrent models between FSMs to model collaboration between system functions.

The Behavioral View

— to validate the consistency of the system behavior:
  ♦ checking of the system modes specification with the Hierarchical Finite States Machine formalism,
  ♦ checking of the delays across the system with the Discrete events formalism.

— The specifier uses these formalisms to figure out how the system must behave and react to external and internal events according to the requirements.

— Other formalisms may be selected to meet, for instance, fault-tolerant constraints.
The Operational View

To validate the completeness of the system behavior
- the missions of the system through scenarios:
  - tactical environment definition,
- the Human/System Interface:
  - symbology,
  - controls (joystick, etc.),
  - visual definitions.
- The customer and the specifier agree on what must be the system behavior, and missions.

The Architectural View

Refinement of the model in terms of components and an architecture:
- components capture technical functions:
  - processing components,
  - control components,
  - sensors and actuators components,
  - etc.
- architecture captures interactions between components:
  - models of concurrency, communication and control.
- definition of component interface with the architecture and avoid direct dependency between components.
- Last stage before technology choices!
Methodology

System Requirements

Operational View

Behavioral View

CoValidation

Operational Architecture

Architectural View

Technical Architecture

The proposed Infrastructure
**Why Ptolemy ?**

— Formalisms composition :
  - formalisms may be composed to model the whole system.

— Several formalisms :
  - DataFlow for processing components,
  - Finite State Machines for control components,
  - Discrete Events for performance evaluation.

— An infrastructure with general purpose services :
  - libraries of reusable components,
  - graphers,
  - interfaces with external tools such as Matlab, Mathematica, etc.
  - openness : source codes available,
  - extendability : object-oriented framework.

---

**Formalisms composition**

<table>
<thead>
<tr>
<th>System S and sub-systems S1, S2, S3</th>
</tr>
</thead>
<tbody>
<tr>
<td>S  ≡  S1</td>
</tr>
</tbody>
</table>

| Models M of S and M1 of S1, M2 of S2, M3 of S3, etc. |

If M is composed with M1, M2 and M3, then M is a model of S.
The Ptolemy DIS domain

- Ptolemy is well-suited to support
  - the Behavioral View of the system model,
  - and the Architectural View.

- We extended Ptolemy to support
  - the Operational View through interfaces with the DIS (Distributed Interactive Simulation) standard protocol from DoD DM&S Office.

- The DIS domain is derived from the DE domain
  - events are dispatched to the inputs of the behavioral model,
  - outputs of the behavioral model emit events to the operation model.

---

Time Management Interoperability

Goal: provide services to support interoperability among federates with different local time management schemes in a single federation execution.

Observation: RTI by itself cannot guarantee interoperability.
Event Driven Federate

Goal: merge TSO messages (events from other federates) with local events so all events are processed in time stamp order

while (simulation still in progress)

invoke Next Event Request ($T_{next} = time stamp of next local event$)

RTI delivers next TSO event w/ time stamp $>$ $T_{local}$, if any exist (+ others w/ same time stamp)

RTI advances federate's logical time, invokes Time Advance Grant

if (TSO message(s) delivered in above Next Event Request service call)

process the remote event(s) delivered to the federate

else

process next local event

Software Architecture

Optronics system control logic

FSM domain

DIS domain

DIS/HLA bus

Visualization MMI (GUI & Man-System Interface)

Tactic model

Carrier model
Conclusion

and

Future Work

---

Conclusion

— Systems definition method improved:
  - DIS simulations to define systems missions,
  - Ptolemy simulations to specify systems supervision,
  - Co-simulations to fill the "gap" between operational and industrial.

— Simulation infrastructure enhanced:
  - Ptolemy DIS domain prototype.

— Maintainability: better changes management
  - help to check introduction of new functions in system definition.

---

Page 16
Future Work

— Refinement of behavioral models with:
  • timing properties, and parametric physical relations or differential equations (CT domain?)
— Add dead-Reckoning to the Ptolemy DIS domain.

Demonstration
The generic Ptolemy DIS model

Display of system reactions

reactions sent to the DIS simulation

DIS Domain

The DIS Target parameters

DE parameters

DIS parameters

OK Apply Close Cancel
THOMSON-CSF OPTRONIQUE

DIS events dispatching

Control Logic
Lase Guided Weapon Mode

Field of View Management
**Synthetic Environment : Approach & Firing**

![Image of synthetic environment](image1)

**Synthetic Environment : Hit and break**

![Image of synthetic environment](image2)
References

- DMSO (Distributed Modeling & Simulation Office):
  - http://www.dmso.mil

- DIS/HLA (Distributed Interactive Simulation/High Level Architecture):
  - http://hla.dmso.mil

- Simulation Based Design Project:
  - http://sbdhost.parl.com

- David Harel:
  - http://www.wisdom.weizmann.ac.il/~harel

- Thomson-CSF Optronique:
  - http://ptolemy.thomson-csf.fr/~warzee