Application of the trace driven process on a Software Radio case-study, experiments and preliminary results

Shuai.Li@fr.thalesgroup.com
Thales Communications & Security
1. Context and objectives
2. Case-study application
3. Application MARTE model
4. Trace generation and analysis
5. Future works
1. Context and objectives

Application Requirements

- Formal modeling of properties
- Software specification
- Detailed design specification

Test scenarios specification

- Business code design
- Wrapper code generation

Software test integration on functional behaviours

Test Trace

Platform Requirements

- Hardware specification
- Platform Design

- Integration from real platform execution

Software components modeling

- Software allocation on hardware, NF constraints and Hw capabilities parameters annotation

Platform modeling

- Functional Verification
- Non-Functional Verification

Trace analysis

Timing comparison execution vs simulation

Integration from real platform execution
1. Context and objectives

**Functional and timing verification**
- Verify arriving order of events and their inter-dependencies
- Verify duration constraints between events

**Models and execution traces**
- Filter execution traces
- Compare with specifications automatically

**Integration and ease-of-use**
- Integrate into current design flow without modifying (too much) system engineer’s habits
- Automate the process through tools development
2. Case-study application

**Time-Division Multiple Access (TDMA)**

<table>
<thead>
<tr>
<th>Frame 0</th>
<th>Frame 1</th>
<th>Frame 2</th>
<th>Frame 3</th>
<th>Frame 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>S0</td>
<td>S1</td>
<td>S2</td>
<td>S3</td>
<td>S4</td>
</tr>
<tr>
<td>S0</td>
<td>S1</td>
<td>S2</td>
<td>S3</td>
<td>S4</td>
</tr>
<tr>
<td>S0</td>
<td>S1</td>
<td>S2</td>
<td>S3</td>
<td>S4</td>
</tr>
<tr>
<td>S0</td>
<td>S1</td>
<td>S2</td>
<td>S3</td>
<td>S4</td>
</tr>
<tr>
<td>S0</td>
<td>S1</td>
<td>S2</td>
<td>S3</td>
<td>S4</td>
</tr>
</tbody>
</table>

Action of station 1

Action of station 2

Action of station 3

**Internal representation**

- **Services** (Time, status, interface, incoming/outgoing data)
- **TDMA Item**
- **Network Controller**

T: Transmission, R: Reception, I: Idle
2. Case-study application

**Software tests implementation**

Activator

- Services
- Network Controller

TDMA Item

Communication by POSIX message queue

**Execution environment**

PandaBoard (OMAP4430) +

Linux 2.6.38.2
3. Application MARTE model
3. Application MARTE model
3. Application MARTE model
3. Application MARTE model
3. Application MARTE model
3. Application MARTE model

<<ClientServerPort>>
3. Application MARTE model

<<ClientServerPort>>
3. Application MARTE model

<<ClientServerPort>>

<<Allocate>>
3. Application MARTE model

<<ClientServerPort>>

<<Allocate>>
3. Application MARTE model
3. Application MARTE model

<<ClientServerPort>>

<<Allocate>>

<<SwSchedulableResource>>
3. Application MARTE model

<<ClientServerPort>>  <<Allocate>>
<<SwSchedulableResource>>
<<MemoryPartition>>
3. Application MARTE model

<<ClientServerPort>>
<<Allocate>>
<<SwSchedulableResource>>
<<MemoryPartition>>
3. Application MARTE model

<<ClientServerPort>>
<<Allocate>>
<<SwSchedulableResource>>
<<MemoryPartition>>
<<StorageResource>>

imProvements of industrial Real time Embedded SysTem s developement prOcess  April 2012
3. Application MARTE model
MARTE stereotypes

3. Application MARTE model

- <<ClientServerPort>>
- <<Allocate>>
- <<SwSchedulableResource>>
- <<MemoryPartition>>
- <<StorageResource>>
- <<Scheduler>>
3. Application MARTE model

<<ClientServerPort>>

<<Allocate>>

<<SwSchedulableResource>>

<<MemoryPartition>>

<<StorageResource>>

<<Scheduler>>
3. Application MARTE model

<<ClientServerPort>>
<<Allocate>>
<<SwSchedulableResource>>
<<MemoryPartition>>
<<StorageResource>>
<<Scheduler>>
<<HwProcessor>>
3. Application MARTE model
4. Trace generation and analysis

**Specification trace**

- Model to text transformation from sequence diagrams:
  - Implemented as an Eclipse plugin
  - Events, dependencies, duration constraints

**Execution trace**

- Use of TAU to obtain execution traces:
  - Code instrumentation:
    - PDT for code analysis
    - TAU for instrumentation and compiling
  - Execution on board for raw traces
  - Filter raw traces and present in a readable format
  - Instant time, function start/end, thread and cpu info

Developed tool to compare execution and specification traces:
- Event order
- Duration between events
5. Future works

**Automatic instrumentation from models**

- Add new stereotypes to ports in order to add instrumentation points automatically in methods of the generated code.

**Data values in execution traces**

- Function parameters for PDU analysis
- Messages sent/received (e.g. structures exchanged by message queue)

**Traces to model**

- Use execution traces to automatically complete the model with quantitative and non-functional properties

**Host and target execution traces exploitation**

- Determine target timing information from host execution traces
Thank you for your attention.

Shuai.Li@fr.thalesgroup.com