COMPASS GRAPHICAL MODELLER

Viet Yen Nguyen

Software Modelling and Verification Group
RWTH Aachen University

Final Presentation Days, April 2012, ESTEC

Christian Dehnert, Joost-Pieter Katoen, Thomas Noll
Pierre Dissaux, Jerome Lagrand, Arnaud Schach
Regis de Ferluc, Gerald Garcia, Xavier Olive
AGENDA

COMPASS

GRAPHICAL MODELLER

EPILOGUE
WHO’S WHO: JOINT INDUSTRIAL-ACADEMIC R&D

Consortium

- RWTH Aachen University
  Software Modeling and Verification Group
- Fondazione Bruno Kessler
  Embedded Systems Group
- Ellidiss Technologies (new)
  STOOD, HOOD, AADL.
- Thales Alenia Space
  World-wide #1 in satellite systems
- European Space Agency
  System Software Engineering Section
EARLY PHASE-B LIMITATIONS ON V&V

Limitation

- HW verified independently of SW with exaggerated mutual assumptions.
- Safety & dependability analyses are isolated from HW/SW models.
- Multiple modeling formalisms for different system aspects (e.g., real-time, probabilistic, hybrid).
- Non-nominal operational modes are overly abstracted to fit various models.
Limitation

HW verified independently of SW with exaggerated mutual assumptions.
EARLY PHASE-B LIMITATIONS ON V&V

Limitation

HW verified independently of SW with exaggerated mutual assumptions.

Safety & dependability analyses are isolated from HW/SW models.
EARLY PHASE-B LIMITATIONS ON V&V

**Limitation**

| HW verified independently of SW with exaggerated mutual assumptions. |
| Safety & dependability analyses are isolated from HW/SW models. |
| Multiple modeling formalisms for different system aspects (e.g. real-time, probabilistic, hybrid). |

**Diagram**

- Hardware
- Software
- System
- Safety/Dependability
- UML
- Simulink
- RtUML
- SysML
- PPT Shapes
- Stochastic Timed Petri Net
- Relex
EARLY PHASE-B LIMITATIONS ON V&V

Limitation

- HW verified independently of SW with exaggerated mutual assumptions.
- Safety & dependability analyses are isolated from HW/SW models.
- Multiple modeling formalisms for different system aspects (e.g. real-time, probabilistic, hybrid).
- Non-nominal operational modes are overly abstracted to fit various models.
Solution: COMPASS Toolset
(since 2008)
<table>
<thead>
<tr>
<th>Model</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>· Architecture, Analysis &amp; Design Language (AADL).</td>
<td></td>
</tr>
<tr>
<td>· Component-oriented, hierarchical modelling.</td>
<td></td>
</tr>
<tr>
<td>· Nominal HW/SW + dynamic reconfiguration.</td>
<td></td>
</tr>
<tr>
<td>· Error behaviour, failure rates from FITs.</td>
<td></td>
</tr>
<tr>
<td>· Fault injections.</td>
<td></td>
</tr>
</tbody>
</table>
## COMPASS TOOLSET 2.2 FEATURES
### MODEL-BASED SYSTEM-SOFTWARE CO-ENGINEERING

<table>
<thead>
<tr>
<th>Model</th>
<th>Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>· Component-oriented, hierarchical modelling.</td>
<td></td>
</tr>
<tr>
<td>· Nominal HW/SW + dynamic reconfiguration.</td>
<td></td>
</tr>
<tr>
<td>· Error behaviour, failure rates from FITs.</td>
<td></td>
</tr>
<tr>
<td>· Fault injections.</td>
<td></td>
</tr>
<tr>
<td>Built upon solid formal methods: automata theory, SAT-solving,</td>
<td></td>
</tr>
<tr>
<td>(probabilistic) temporal logics &amp; (probabilistic) model checking.</td>
<td></td>
</tr>
</tbody>
</table>

2012, Viet Yen Nguyen
# COMPASS TOOLSET 2.2 FEATURES
## MODEL-BASED SYSTEM-SOFTWARE CO-ENGINEERING

<table>
<thead>
<tr>
<th>Model</th>
<th>Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td>· Component-oriented, hierarchical modelling.</td>
<td></td>
</tr>
<tr>
<td>· Nominal HW/SW + dynamic reconfiguration.</td>
<td></td>
</tr>
<tr>
<td>· Error behaviour, failure rates from FITs.</td>
<td></td>
</tr>
<tr>
<td>· Fault injections.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Analyses</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>· Simulation.</td>
<td>(model checking)</td>
</tr>
<tr>
<td>· Functional correctness.</td>
<td>(safety &amp; dependability)</td>
</tr>
<tr>
<td>· Fault tree generation.</td>
<td>(safety &amp; dependability)</td>
</tr>
<tr>
<td>· FMEA generation.</td>
<td>(safety &amp; dependability)</td>
</tr>
<tr>
<td>· Fault tolerance evaluation.</td>
<td>(safety &amp; dependability)</td>
</tr>
<tr>
<td>· Fault tolerance evaluation.</td>
<td>(safety &amp; dependability)</td>
</tr>
<tr>
<td>· Fault detection analysis.</td>
<td>(FDIR effectiveness)</td>
</tr>
<tr>
<td>· Fault isolation analysis.</td>
<td>(FDIR effectiveness)</td>
</tr>
<tr>
<td>· Failure recovery analysis.</td>
<td>(FDIR effectiveness)</td>
</tr>
<tr>
<td>· Diagnosability.</td>
<td>(FDIR effectiveness)</td>
</tr>
<tr>
<td>· Fault tree evaluation.</td>
<td>(Probabilistic risk assessment)</td>
</tr>
<tr>
<td>· Performability.</td>
<td>(Probabilistic risk assessment)</td>
</tr>
</tbody>
</table>
# COMPASS TOOLSET 2.2 FEATURES

## MODEL-BASED SYSTEM-SOFTWARE CO-ENGINEERING

| Model | · Architecture, Analysis & Design Language (AADL).  
· Component-oriented, hierarchical modelling.  
· Nominal HW/SW + dynamic reconfiguration.  
· Error behaviour, failure rates from FITs.  
· Fault injections. |
| Requirements | · Pattern-based, functional & probabilistic. |
| Analyses | · Simulation.  
· Functional correctness. (model checking)  
· Fault tree generation. (safety & dependability)  
· FMEA generation. (safety & dependability)  
· Fault tolerance evaluation. (safety & dependability)  
· Failure detection analysis. (FDIR effectiveness)  
· Fault isolation analysis. (FDIR effectiveness)  
· Failure recovery analysis. (FDIR effectiveness)  
· Diagnosability. (FDIR effectiveness)  
· Fault tree evaluation. (Probabilistic risk assessment)  
· Performability. (Probabilistic risk assessment) |

Built upon solid formal methods: automata theory, SAT-solving, (probabilistic) temporal logics & (probabilistic) model checking.
EXAMPLE MODEL REDUNDANT SENSOR-FILTER SYSTEM
COMPONENT TYPES DESCRIBE WHAT TO COMMUNICATE

```plaintext
system Acquisition
  features
    value: out data port int default 4;
    alarmS: out data port bool default false observable;
    alarmF: out data port bool default false observable;
end Acquisition;
```

2012, Viet Yen Nguyen
EXAMPLE MODEL REDUNDANT SENSOR-FILTER SYSTEM
COMPONENT IMPLEMENTATIONS DESCRIBE HOW TO COMMUNICATE

```
system Acquisition
  features
    value: out data port int default 4;
    alarmS: out data port bool default false observable;
    alarmF: out data port bool default false observable;
end Acquisition;

system implementation Acquisition.Impl
  subcomponents
    sensors: system Sensors accesses mybus;
    filters: system Filters accesses mybus;
    monitor: system Monitor accesses mybus;
    mybus: bus MyBus;
  connections
    data port sensors.output -> filters.input;
    data port filters.output -> value;
    data port sensors.output -> monitor.valueS;
    data port filters.output -> monitor.valueF;
    data port monitor.alarmS -> alarmS;
    data port monitor.alarmF -> alarmF;
    event port monitor.switchS -> sensors.switch;
    event port monitor.switchF -> filters.switch;
end Acquisition.Impl;
```
EXAMPLE MODEL REDUNDANT SENSOR-FILTER SYSTEM

```
system Sensors
  features
    output: out data port int default 1;
    switch: in event port;
  end Sensors;
```
EXAMPLE MODEL REDUNDANT SENSOR-FILTER SYSTEM
MODE TRANSITION SYSTEM GIVES RISE TO DYNAMIC BEHAVIOUR

```plaintext
system Sensors
  features
    output: out data port int default 1;
    switch: in event port;
end Sensors;

system implementation Sensors.Impl
  subcomponents
    sensor1: device Sensor in modes (Primary);
    sensor2: device Sensor in modes (Backup);
  connections
    data port sensor1.output -> output in modes (Primary);
    data port sensor2.output -> output in modes (Backup);
  modes
    Primary: activation mode;
    Backup: mode;
  transitions
    Primary -[switch]-> Backup;
end Sensors.Impl;
```
Pierre takes over.
The resulting FMEA table is presented **underneath**

<table>
<thead>
<tr>
<th>Num</th>
<th>Failure Model</th>
<th>Failure Effect</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>filters.filter1._errorSubcomponent.#die = 1</td>
<td>(value &gt;= 15</td>
</tr>
<tr>
<td>2</td>
<td>sensors.sensor1._errorSubcomponent.#die = 1</td>
<td>(value &gt;= 15</td>
</tr>
<tr>
<td>3</td>
<td>sensors.sensor1._errorSubcomponent.#die = 1</td>
<td>sensors.sensor1.output &gt;= 15</td>
</tr>
</tbody>
</table>
COMPASS TOOLSET SCREENSHOTS

PERFORMABILITY

Properties

Name

- Sensorbank die before 1 year
- Second sensor dies before 2 years

Cumulative Distribution Function

Settings

Maximum Probability: 87.2%
COMPASS TOOLSET SCREENSHOTS

DIAGNOSABILITY

Trace A
Compliant with diagnosability property.

Trace B
Not compliant with diagnosability property.

---

Name | Step1 | Step2 | Step3 | Step4
---|---|---|---|---
mode | | | | |
monitor.detected | OK | OK | OK | OK
monitor.mode | | | | |
monitor.switchS | 0 | 15 | 15 | 15
monitor.valueS | | | | |
mybus.mode | | | | |
sensors.mode | | | | |
sensors.output | | | | |
sensors.sensor1.#die | | | | |
sensors.sensor1.#dieByDrift1 | | | | |
sensors.sensor1.#dieByDrift2 | | | | |
sensors.sensor1.#dieByGlitch | | | | |
sensors.sensor1.#drift1 | | | | |
sensors.sensor1.#drift2 | OK | Drifted1 | ...rifted... | |
sensors.sensor1.#error | | | | |
sensors.sensor1.#glitch | | | | |
sensors.sensor1.mode | Cycle | Cycle | Cycle | ...

Name | Step1 | Step2 | Step3 | Step4
---|---|---|---|---
mode | | | | |
monitor.detected | OK | OK | OK | OK
monitor.mode | | | | |
monitor.switchS | 0 | 0 | 0 | 0
monitor.valueS | | | | |
mybus.mode | | | | |
sensors.mode | | | | |
sensors.output | | | | |
sensors.sensor1.#die | | | | |
sensors.sensor1.#dieByDrift1 | | | | |
sensors.sensor1.#dieByDrift2 | | | | |
sensors.sensor1.#dieByGlitch | | | | |
sensors.sensor1.#drift1 | | | | |
sensors.sensor1.#drift2 | OK | ...rifted... | Drifted1 | |
sensors.sensor1.#error | | | | |
sensors.sensor1.#glitch | | | | |
sensors.sensor1.mode | Cycle | Cycle | Cycle | ...

---

2012, Viet Yen Nguyen
EVALUATION

CGM: by Thales through modelling subsystems of a satellite.

+ Intuitive graphical notation, useful for documentation.
+ Mature and user-friendly tool.
+ Models directly analysable by COMPASS.
+ Supports both top-down (system first) and bottom-up (units first) modelling.
+ Possibility of mapping CCM/SysML/CHESS for visualisation.
CGM: by Thales through modelling subsystems of a satellite.

+ Intuitive graphical notation, useful for documentation.
+ Mature and user-friendly tool.
+ Models directly analysable by COMPASS.
+ Supports both top-down (system first) and bottom-up (units first) modelling.
+ Possibility of mapping CCM/SysML/CHESS for visualisation.

/ Multiple diagram support.
/ Enhanced copy and paste by selections.
/ Enhanced algorithms for automatic layouting.
EVALUATION

CGM: by Thales through modelling subsystems of a satellite.
+ Intuitive graphical notation, useful for documentation.
+ Mature and user-friendly tool.
+ Models directly analysable by COMPASS.
+ Supports both top-down (system first) and bottom-up (units first) modelling.
+ Possibility of mapping CCM/SysML/CHESS for visualisation.
  / Multiple diagram support.
  / Enhanced copy and paste by selections.
  / Enhanced algorithms for automatic layouting.

COMPASS: many case studies, a full satellite platform is the largest:
• ICSE’12: *Formal Correctness, Safety, Dependability and Performance Analysis of a Satellite.*
• ERTSS’10: *Formal Verification and Validation of AADL Models.*
The COMPASS Graphical Modeller is an user-friendly drag-'n-drop modelling tool of system hierarchy and behaviour.

Link to COMPASS for model-based V&V by correctness, safety, dependability, performance, FDIR analysis.
CONCLUSION

The COMPASS Graphical Modeller is an user-friendly drag-'n-drop modelling tool of system hierarchy and behaviour.

Link to COMPASS for model-based V&V by correctness, safety, dependability, performance, FDIR analysis.

More information & availability:

- COMPASS Toolset: community open source for companies and citizens of ESA member states.
- COMPASS Graphical modeller: free for non-commercial use. For commercial use, contact Ellidiss.
- Website: http://compass.informatik.rwth-aachen.de/