SATELLITE PLATFORM CASE STUDY WITH SLIM AND COMPASS

Viet Yen Nguyen

Joint work with Marie-Aude Esteve, Joost-Pieter Katoen, Bart Postma and Yuri Yushtein.
OUR CASE: SATELLITE PLATFORM

- From ESA mission in development.
- Platform keeps satellite in space, like car’s chassis.
  - control & data unit,
  - propulsion,
  - telemetry, tracking & cmd.
  - power,
  - attitude & orbit control sys.,
  - reconfiguration module,
  - ...
- Fault detection, isolation, recovery (FDIR):
  - redundancies + recovery,
  - compensation algorithms,
  - failure isolation schemes,
  - omnipresent in satellite.

Note: shown satellite is not the one of the case study.
DEMANDING REQUIREMENTS NEED EXTENSIVE V&V

• “Satellite’s in-orbit lifetime shall be at least 5 years.”
• “Satellite’s platform reliability shall be equivalent or less than 1000 FIT.”
• “The probability that the satellite transits to safe mode shall be less than 0.001, considering 20 hours for ground time to repair.”
• “In case of an over-current, a switch to the redundant coil is performed by OBDH SW.”
• … thousands of similar requirements!
• In presence of space hazards like radiation, extreme heat/cold, radio latencies, gravitational forces, etc.

Note: shown requirements are not from the case study.
4.4 Project phasing

4.4.1 Introduction

The life cycle of space projects is typically divided into 7 phases, as follows:

<table>
<thead>
<tr>
<th>Phase</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Mission analysis/needs identification</td>
</tr>
<tr>
<td>A</td>
<td>Feasibility</td>
</tr>
<tr>
<td>B</td>
<td>Preliminary definition</td>
</tr>
<tr>
<td>C</td>
<td>Detailed definition</td>
</tr>
<tr>
<td>D</td>
<td>Qualification and production</td>
</tr>
<tr>
<td>E</td>
<td>Utilization</td>
</tr>
<tr>
<td>F</td>
<td>Disposal</td>
</tr>
</tbody>
</table>

A typical project life cycle is illustrated in Figure 4-3.

Activities are closely linked to activities on system and product level. Depending on the specific circumstances of a project and the acceptance of involved risk, activities can overlap project phases. At the conclusion of the major activities and the related project reviews, configuration baselines are established (see ECSS-M-ST-40).

- Requirements analysis.
- Fault tree analysis.
- Review and inspection.
- ... 

- HW/SW co-testing.
- Early prototyping.
- Control simulation.
- ...

- End-to-end IS testing.
- Operations readiness.
- Mission scenario tests.
- ...

- In orbit testing.

---

CONTINUOUS V&V FOR RISK/BUDGET/PLANNING
CHALLENGES IN SPACE VERIFICATION & VALIDATION

- More demanding requirements.
- More FDIR software.
- More system behaviors.
- More issues during V&V
- More pressure on meeting cost and schedule constraints!

Example: once every 175 year launch window for Voyager 2.
OUR SOLUTION: COMPASS
ARCHITECTURE & ANALYSIS DESIGN LANGUAGE: ONE LANGUAGE FOR ALL SYSTEM-LEVEL ASPECTS

• Standardized by SAE.
• Component-oriented and hierarchical.
• HW (processors, devices, bus, etc.).
• SW (threads, etc.).
• Modes and mode transitions.
• Communication by event data port connections.
• Dynamic reconfiguration.
• Error events and error states.
CASE STUDY MODEL

Detailed version of this is confidential.
# STUDY PARALLEL WITH SATELLITE DEVELOPMENT

<table>
<thead>
<tr>
<th>Activities</th>
<th>Phase 0</th>
<th>Phase A</th>
<th>Phase B</th>
<th>Phase C</th>
<th>Phase D</th>
<th>Phase E</th>
<th>Phase F</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mission/Function</td>
<td>MDR</td>
<td>PRR</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Requirements</td>
<td></td>
<td></td>
<td>SRR</td>
<td>PDR</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- One modeler.
  - No prior experience with space sys. engineering.
  - Basic model checking.
- One supporter (me).
- Six months duration.
  - 2 mo. working in.
  - 4 mo. model & analysis.
- Activity in parallel with satellite development:
  - No fully mature design.
  - Details need to be decided.

Another case study run on the same mission, but for CDR. Not part of this presentation.
OUR CASE: SATELLITE PLATFORM

- From ESA mission in development.
- Platform keeps satellite in space, like car’s chassis.
  - control & data unit,
  - propulsion,
  - telemetry, tracking & cmd.
  - power,
  - attitude & orbit control sys.,
  - reconfiguration module,
  - ...
- Fault detection, isolation, recovery (FDIR):
  - redundancies + recovery,
  - compensation algorithms,
  - failure isolation schemes,
  - omnipresent in satellite.

Note: shown satellite is not the one of the case study.
WHAT’S MODELED: ARCHITECTURE AND FDIR OPS.  
(THIS LIST IS NOT EXHAUSTIVE)

Most particularly:
• Voting algorithms on hot redundant elements.
• Off-range checkers.
• Initialization sequences.
• Recovery sequences, disabling/enabling system elements.
• Flags:
  • Status
  • Commandability
• Failure configurations.
• Evolution of physical quantities:
  • Time
  • Pressure
  • Heat
• Timing constraints.
# Largest System-Level Model Ever Developed

<table>
<thead>
<tr>
<th>Scope</th>
<th>Metric</th>
<th>Count</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model</td>
<td>Components</td>
<td>86</td>
</tr>
<tr>
<td></td>
<td>Ports</td>
<td>937</td>
</tr>
<tr>
<td></td>
<td>Modes</td>
<td>244</td>
</tr>
<tr>
<td></td>
<td>Error models</td>
<td>20</td>
</tr>
<tr>
<td></td>
<td>Recoveries</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>Nominal state space</td>
<td>48421100</td>
</tr>
<tr>
<td></td>
<td>LOC (without comments)</td>
<td>3831</td>
</tr>
<tr>
<td>Requirements</td>
<td>Propositional</td>
<td>25</td>
</tr>
<tr>
<td></td>
<td>Absence</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Universality</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Response</td>
<td>14</td>
</tr>
<tr>
<td></td>
<td>Probabilistic Invariance</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>Probabilistic Existence</td>
<td>1</td>
</tr>
</tbody>
</table>
All analysis results are in the publication.

CASE STUDY ANALYSES
INJECTING FAULTS EXPLODE THE STATE SPACE

- Single earth sensor signal failure: 3 injections, 1 fault
- Double earth sensors signal failure: 11 injections, 2 faults
- Propulsion failure: 35 injections, 5 faults
- AOCS equipments failure: 213563 injections, 7 faults
- All reactionwheel failures: 16 injections, 8 faults
- Single reactionwheel failure: 10 injections, 10 faults
- Processor module failures: 1372 injections, 10 faults
- Complete earth sensor failure: 11 injections, 14 faults
- Reactionwheel + earth sensor failures: 172 injections, 22 faults

Explosion correlates with scope of failure.
EXHAUSTIVE VERIFICATION WITH MODEL CHECKING

- Push-button technology. Requires tractable defaults.
- Counterexample upon violation of property/requirement.
AUTOMATIC GENERATION OF FAULT TREES

• In space engineering practice done manually!
• Relates effects to failures.
• Full generation for safe mode transition. 66 nodes.
• Priority-AND gates (dynamic FT) for more precision.
AUTOMATIC PROBABILISTIC RISK ASSESSMENT

• Transformation to continuous time Markov chain.
• Computes probabilities of events and gates firing.
• Supports quantified decision making.
FAILURE MODES AND EFFECTS TABLE GENERATION

- In space engineering practice done manually!
- Relates failures to effects. Opposite of fault trees.
- Increase cardinality to discover new fault configurations.
INTRACTABLE DUE TO ALGORITHMIC COMPLEXITY

Diagnosability
- Checks sufficiency of observables.
- Based on TwinPlant reachability.

Performability
- Reliability, availability, maintainability.
- Based on stochastic weak bisimulation of interactive Markov chains.
LIST OF ALL COMPASS 2.2 ANALYSES

• Requirements validation.
• Random, user-guided, constraint simulation.
• Deadlock detection.
• (Hybrid) **Model checking** using BDD/SMT.
• (Dynamic) **Fault tree generation**.
• (Dynamic) **Fault tree evaluation**.
• (Dynamic) Fault tree verification.
• Fault tolerance evaluation.
• (Dynamic/Compact) **FMEA**.
• Fault detection analysis.
• Fault isolation analysis.
• Fault recovery analysis.
• **Diagnosability** analysis.
• **Performability**.
thus avoiding increased costs. In our case study, we detected early, and have them resolved long before integration testing, formally forces engineers to consider design issues outputs that improve the eventual system under development. Generally understood that formal modeling and analysis provides more generic. This case study, but could be further developed to become redundant configurations. The patterns now tailored to occurring concepts, like recovery procedures and particular number of architectural patterns for modeling frequently of them is impractical.

These two manual checks are tedious, especially given a possible infinite trace, all clocks are reset at least once. Divergence is avoided by checking manually that on each loop consuming time, hence avoiding Zeno behavior. Time positive. This ensures there is at least one event on each

<table>
<thead>
<tr>
<th>Analysis</th>
<th>Fault Injections</th>
<th>Properties</th>
<th>Time (sec)</th>
<th>Memory (MB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Discrete model checking</td>
<td>(none, i.e. nominal behavior only)</td>
<td>“health check on valves is performed” and “no firing of thrusters triggers reconfiguration” and “thrusters not stopping firing triggers reconfiguration” and “overpressure triggers opening latch valve”</td>
<td>224</td>
<td>122</td>
</tr>
<tr>
<td>Hybrid model checking (10*)</td>
<td>Single Earth sensor signal failure</td>
<td>“No thruster usage during nominal operation”</td>
<td>23</td>
<td>242</td>
</tr>
<tr>
<td>Hybrid model checking (20*)</td>
<td>Single Earth sensor signal failure</td>
<td>“No thruster usage during nominal operation”</td>
<td>52</td>
<td>360</td>
</tr>
<tr>
<td>Hybrid model checking (30*)</td>
<td>Single Earth sensor signal failure</td>
<td>“No thruster usage during nominal operation”</td>
<td>101</td>
<td>492</td>
</tr>
<tr>
<td>Hybrid model checking (40*)</td>
<td>Single Earth sensor signal failure</td>
<td>“No thruster usage during nominal operation”</td>
<td>204</td>
<td>612</td>
</tr>
<tr>
<td>Hybrid model checking (50*)</td>
<td>Single Earth sensor signal failure</td>
<td>“No thruster usage during nominal operation”</td>
<td>361</td>
<td>713</td>
</tr>
<tr>
<td>Hybrid model checking (60*)</td>
<td>Single Earth sensor signal failure</td>
<td>“No thruster usage during nominal operation”</td>
<td>967</td>
<td>884</td>
</tr>
<tr>
<td>Hybrid model checking (70*)</td>
<td>Single Earth sensor signal failure</td>
<td>“No thruster usage during nominal operation”</td>
<td>2176</td>
<td>1006</td>
</tr>
<tr>
<td>Fault tree analysis</td>
<td>Double Earth sensors signal failure</td>
<td>i.d.</td>
<td>555</td>
<td>134</td>
</tr>
<tr>
<td>Fault tree analysis</td>
<td>AOCs equipments failure</td>
<td>TLE-1</td>
<td>2898</td>
<td>181</td>
</tr>
<tr>
<td>Fault tree analysis</td>
<td>Double Earth sensors signal failure</td>
<td>“fail-operational flag is set”</td>
<td>769</td>
<td>132</td>
</tr>
<tr>
<td>Fault tree analysis</td>
<td>Processor module failures</td>
<td>“CDU alarms are raised”</td>
<td>483</td>
<td>134</td>
</tr>
<tr>
<td>Fault tree analysis</td>
<td>AOCs equipments failure</td>
<td>“fail-operational flag is set”</td>
<td>8349</td>
<td>239</td>
</tr>
<tr>
<td>Dynamic fault tree analysis</td>
<td>Double Earth sensors signal failure</td>
<td>“fail-operational flag is set”</td>
<td>630</td>
<td>135</td>
</tr>
<tr>
<td>Dynamic fault tree analysis</td>
<td>Processor module failures</td>
<td>“CDU alarms are raised”</td>
<td>547</td>
<td>136</td>
</tr>
<tr>
<td>Dynamic fault tree analysis</td>
<td>AOCs equipments failure</td>
<td>“fail-operational flag is set”</td>
<td>5581</td>
<td>212</td>
</tr>
<tr>
<td>FMEA table generation</td>
<td>Double Earth sensor signal failure</td>
<td>“fail-operational flag is set”</td>
<td>1003</td>
<td>134</td>
</tr>
<tr>
<td>Fault detection analysis</td>
<td>Double Earth sensor signal failure</td>
<td>TLE-1</td>
<td>1173</td>
<td>142</td>
</tr>
<tr>
<td>Fault isolation analysis</td>
<td>Double Earth sensor signal failure</td>
<td>n.a.</td>
<td>21920</td>
<td>136</td>
</tr>
<tr>
<td>Diagnosability analysis</td>
<td>Double Earth sensor signal failure</td>
<td>TLE-1</td>
<td>586093</td>
<td>1474</td>
</tr>
<tr>
<td>Performability</td>
<td>Single Earth sensor signal failure</td>
<td>TLE-1</td>
<td>33166</td>
<td>2103</td>
</tr>
<tr>
<td>Fault tree evaluation</td>
<td>Double Earth sensor signal failure</td>
<td>“fail-operational flag is set”</td>
<td>1</td>
<td>n.a.</td>
</tr>
<tr>
<td>Dynamic fault tree evaluation</td>
<td>Double Earth sensor signal failure</td>
<td>“fail-operational flag is set”</td>
<td>1</td>
<td>n.a.</td>
</tr>
</tbody>
</table>
Been there, done that.

WHAT’S NEXT?
1. MODEL CONSTRUCTION PROBLEM REVISITED

Many documents:
• Design
• Requirements
• Analysis reports
• Justifications
• Procedures
• Summaries

Varying fidelities & maturities of system/SW artifacts.

How to put all this into a model?
1. MODEL CONSTRUCTION PROBLEM REVISITED

• For analysis, equalize model/information fidelities.
• Continuous abstraction-levels!

\[ \text{Req’s} = | \text{model} = | \text{system}. \]

Theory (computation) model experiment.

- use of “how” vs “what” requirements blur during detailed design.
- your opinion of DOORS-style requirements?
- DO-178C, DO-331 Modeling supplement?
2. EXPLORATION VS CHECKING (FORMAL) METHODS

<table>
<thead>
<tr>
<th>Activities</th>
<th>Phases</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Phase 0</td>
</tr>
<tr>
<td>Mission/Function</td>
<td>Explorative Methods</td>
</tr>
<tr>
<td>Requirements</td>
<td></td>
</tr>
<tr>
<td>Definition</td>
<td>COMPASS</td>
</tr>
<tr>
<td>Verification</td>
<td>TASTE</td>
</tr>
<tr>
<td>Production</td>
<td></td>
</tr>
<tr>
<td>Utilization</td>
<td></td>
</tr>
</tbody>
</table>

Appropriate model. Appropriate analysis methods. At each engineering phase.
SPECIFICALLY…

Random Trends

Improving COMPASS
3. NOT SO RANDOM TRENDS

• Software failures. SFMECA, SFTA.
  – Assumption that SW never fails is wrong.
  – How does SW fail?
  – When to handle SW faults? (design vs. runtime)
  – ECSS-Q-HB-80-03A: “Software Dependability and Safety”

• Ontology supported modeling & analysis.
  – Increasingly more databases, taxonomies, and relations.

• Fault propagation. Fault management.
  – Somehow, questions (S)FMECA vs. (S)FTA.

• Measures for accounting.
  – Project/Cost/* manager need to budget.
  – A priori state space estimation for alg. time/memory complexity.
4. COMPASS-SPECIFIC TODO LIST

- Improving diagnosability.
  - Better algorithms.
  - Handling delayed diagnosis.
  - Go beyond Boolean diagnosis.

- Improving FMEA and FTA generation.
  - Generate layered FTA according to system hierarchy.
  - Likewise for FMEA. Strengthen connection with FTA.

- Improving Probabilistic Risk Assessment.
  - Better algorithms.
  - Combine continuous Markov-model with real-timed transitions.
  - Devise underlying formal Markov model for reactive systems.

- More in my PhD thesis.
CONCLUSION
MODEL, ANALYSES, REPORT ON SATELLITE CASE

- Space sys/sw co-engineering.
- Rigorous verification & validation.

- Demanding requirements.
- More FDIR SW.
- SW ←→ system perspective.

- AADL
- Largest formal system model.
- Real-time, hybrid, probabilistic.

European Space Agency

Satellite

- Generate V&V artifacts.
- (Probabilistic) model checking.
- Up to +/- 500 million states.

Analyses

Model

- Best practices.
- Lessons learned.
- Integration in engineering process.

Report
MORE INFORMATION


• COMPASS Toolset: compass.informatik.rwth-aachen.de/