Automated Failure-modes-and-effects Analysis of Embedded Software

Peter Struss
Technische Universität München
struss@in.tum.de
Preliminaries

Disclaimer

• Software FMEA – but
  – not a software-centered perspective
• FMEA of embedded systems –
  – not a general proposal for SW FMEA
• Solid foundation and some experiments
  – But still work in progress (early stage)

Claim

• Reason for feasibility and strength of the approach
  – not a software-centered perspective
  – not a general proposal for SW FMEA
• Different from other approaches to FMEA of embedded systems (e.g. Snooke-Price RAMS 2011)
FMEA of Embedded Systems - Idea

- Contexts?
- Granularity?
- Output distinctions?

- What the software does (wrong)
  - is irrelevant!
  - unless affecting the behavior of the physical system
The Physical System Dictates the Software Analysis

• Which contexts?
• Which granularity?
• Which output distinctions?

• Physical system
  – determines and restricts the context
  – Determines relevant distinctions
• FMEA
  – purely qualitative
  – provides finite set of relevant effects
Outline

- Model-based FMEA – Foundations
- Models for FMEA of a Braking System
- FMEA of Cyber-physical Systems
Model-based FMEA - Foundations
The FMEA Task

Task:
- During design phase:
- Determine of the effects of (classes of) component faults
- under different operating conditions
FMEA Table – Simple Example

- Manual composition of a FMEA
- Case: Pipe 1
- → A **qualitative** analysis!

<table>
<thead>
<tr>
<th>ITEM</th>
<th>Function</th>
<th>FAILURE MODE</th>
<th>Mission Phase</th>
<th>LOCAL EFFECT</th>
<th>Next Higher Level Effect</th>
<th>END EFFECT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pipe 1</td>
<td>Transport fluid</td>
<td>Big Leakage</td>
<td>Descent (cmd=1)</td>
<td>Expected flow not produced</td>
<td>No pressure generation</td>
<td>No Extension</td>
</tr>
</tbody>
</table>

- **ITEM:** Pipe 1
- **Function:** Transport fluid
- **FAILURE MODE:** Big Leakage
- **Mission Phase:** Descent (cmd=1)
- **LOCAL EFFECT:** Expected flow not produced
- **Next Higher Level Effect:** No pressure generation
- **END EFFECT:** No Extension
The FMEA Task

Task:
- During design phase:
- Determine of the effects of (classes of) component faults
- under different operating conditions
Task:
• Given a blue print
• Predict the behavior
• of a model of the faulty system
The Core Of The Task

Scenario

Model

Failure Mode

Effect
Checking Consistency, Entailment

Model → Scenario → Failure Mode
                      ↑
Effect                  ↓
                      ↑
Model                  Effect

Effect entailed? consistent?
Model-based Computation of Effects

- Scenario relation $S$
- Failure mode relation $FM_1$
- Effect relations $E_1$, $E_2$, and $E_3$

- Failure mode under scenario: $FM_1 \triangleright S$

- Failure mode included in effect: effect will **definitely** occur: $FM_1 \triangleright S \subseteq E_1$

- Intersection empty: effect **does not** occur: $FM_1 \triangleright S \cap E_2 = \emptyset$

- Otherwise, effect **may** occur: $E_3$
FMEA: Analysis of Qualitative Deviations

Deviations
\[ \Delta x := [x_{act} - x_{ref}] \]

Equations
\[ Q_1 + Q_2 = 0 \]

Model Fragments
\[ \Delta Q_1 + \Delta Q_2 = 0 \]

- \[ \Delta(x + y) = \Delta x + \Delta y \]
- \[ \Delta(x - y) = \Delta x - \Delta y \]
- \[ \Delta(x \cdot y) = x_{act} \cdot \Delta y + y_{act} \cdot \Delta x - \Delta x \cdot \Delta y \]
- \[ \Delta(x / y) = (y_{act} \cdot \Delta x - x_{act} \cdot \Delta y) / (y_{act} \cdot (y_{act} \cdot \Delta y)) \]
- \[ y = f(x) \text{ monotonic } \Rightarrow \Delta x = \Delta y \]
- Reference can be unspecified!
Models for FMEA of a Braking System
Demonstrator: Standard Braking System (BMW)
Model – Actuator Part

VI11

VI21

VI12

PA_C2

PA_P2

PA_C1

PA_P1

J1

J5
Modeling of Hydraulic Components – Modeling Goal

- Qualitative model
- capturing the initial response
- in terms of deviations

- Three p values: 0, (+), +
- Constraints on
  - Magnitude
  - Deviations
  - Integration
  - Persistence
- Details in [Struss-Fraracci QR 11]
Modeling of Hydraulic Components – Initial Response

Vol_{j+1} \rightarrow \text{Valve}_{j} \rightarrow \text{Vol}_3 \rightarrow \text{Valve}_2 \rightarrow \text{Vol}_1

P \quad Q \quad P \quad Q \quad P \quad Q \quad P

S_0 \downarrow \quad (+) \quad \quad 0 \quad \quad +

S_1 \quad (+) \quad \Rightarrow \quad (+) \quad \Leftarrow \quad (+)
## Elements of the Model – Base Model

<table>
<thead>
<tr>
<th></th>
<th>Valve</th>
<th>Volume</th>
</tr>
</thead>
<tbody>
<tr>
<td>Base model</td>
<td>$T_1.Q = A^* (T_1.P - T_2.P)$</td>
<td>$T_1.Q = \partial P$</td>
</tr>
<tr>
<td></td>
<td>$T_1.Q = -T_2.Q$</td>
<td></td>
</tr>
</tbody>
</table>

- $T_1$: Pressure
- $Q$: Flow
- $P$: Pressure
Elements of the Model – Base Model and Integration

<table>
<thead>
<tr>
<th></th>
<th>Valve</th>
<th>Volume</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Base model</strong></td>
<td>$T_1.Q = A^* (T_1.P-T_2.P)$</td>
<td>$T_1.Q = \partial P$</td>
</tr>
<tr>
<td></td>
<td>$T_1.Q = -T_2.Q$</td>
<td></td>
</tr>
<tr>
<td><strong>Continuity</strong></td>
<td><strong>Integration</strong></td>
<td><strong>Persistence</strong></td>
</tr>
<tr>
<td></td>
<td>$Q_0$</td>
<td>$\partial Q$</td>
</tr>
<tr>
<td></td>
<td>$-$</td>
<td>$*$</td>
</tr>
<tr>
<td></td>
<td>$0$</td>
<td>$-$</td>
</tr>
<tr>
<td></td>
<td>$0$</td>
<td>$0$</td>
</tr>
<tr>
<td></td>
<td>$0$</td>
<td>$+$</td>
</tr>
<tr>
<td></td>
<td>$+$</td>
<td>$*$</td>
</tr>
</tbody>
</table>
## Elements of the Model – Derivative Layer

<table>
<thead>
<tr>
<th>Valve</th>
<th>Volume</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Base model</strong></td>
<td><strong>Volume</strong></td>
</tr>
<tr>
<td>[ T_1.Q = A^* (T_1.P - T_2.P) ]</td>
<td>[ T_1.Q = \partial P ]</td>
</tr>
<tr>
<td>[ T_1.Q = -T_2.Q ]</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Continuity</strong></th>
<th><strong>Integration</strong></th>
<th><strong>Persistence</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>( Q_0 )</td>
<td>( \partial Q )</td>
<td>( Q )</td>
</tr>
<tr>
<td>-</td>
<td>*</td>
<td>-</td>
</tr>
<tr>
<td>0</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>(+)</td>
<td>*</td>
<td>(+)</td>
</tr>
<tr>
<td>0</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>+</td>
<td>*</td>
<td>+</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Base model derivative</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>[ T_1.\partial Q = A^* (T_1.\partial P-T_2.\partial P) ]</td>
</tr>
<tr>
<td>[ T_1.\partial Q = -T_2.\partial Q ]</td>
</tr>
</tbody>
</table>
## Elements of the Model – Base Model and Deviation Model

<table>
<thead>
<tr>
<th></th>
<th>Valve</th>
<th>Volume</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Base model</strong></td>
<td>$T_1.Q = A^* (T_1.P - T_2.P)$</td>
<td>$T_1.Q = \partial P$</td>
</tr>
<tr>
<td></td>
<td>$T_1.Q = -T_2.Q$</td>
<td></td>
</tr>
<tr>
<td><strong>Deviation model</strong></td>
<td>$T_1.\Delta Q = \Delta A^* P_{\text{diff}} + A^* \Delta P_{\text{diff}} - A^* \Delta P_{\text{diff}}$</td>
<td>$T_1.\Delta Q = \Delta \partial P$</td>
</tr>
<tr>
<td></td>
<td>$P_{\text{diff}} = T_1.P - T_2.P$</td>
<td></td>
</tr>
<tr>
<td></td>
<td>$T_1.\Delta Q = -T_2.\Delta Q$</td>
<td></td>
</tr>
<tr>
<td><strong>Integration Deviation</strong></td>
<td>$T_i. \Delta \partial Q = T_i. \Delta Q$</td>
<td>$\Delta P = \Delta \partial P$</td>
</tr>
</tbody>
</table>
Sample Inference (Simplified)

- VI 11 not properly open (e.g. stuck-closed)
  - $\Delta A = -$  
  - $\Delta Q = -$  
  - ...  
  - $\Delta P = -$  
  - $\rightarrow$ WB 11 underbraked
FMEA of Cyber-physical Systems
Sensor Failures
Limitations for Systems with Embedded Software

- Sensor failures cannot be analyzed
  - Affects system only via software
- Failures of crucial components: software
- \( \rightarrow \) extend FMEA to embedded software
Propagating Sensor Failures

- E.g. sensor signal too low
- ValveControl_ij OK model
  $\Delta cmd = - \Delta s$
- InletValve_ij OK model
  $\Delta A = - \Delta cmd$

$\Delta s = -$ $\Delta cmd = +$ $\Delta A = -$
FMEA of Cyber-physical Systems
Software Failures
Effect of Software Failures

- E.g. ValveControl untimely signal
  - E.g. threshold too high
  - Permanent signal
- Fault model (single fault)
  $\Delta cmd = +$
- Or $\Delta cmd = - \Delta s \oplus +$

Speed
sensor_ij

Valve
Control_ij

Inlet
Valve_ij

$\Delta s = 0$

$\Delta cmd = +$

$\Delta A = -$
FMEA of Cyber-physical Systems
Refined Analysis
Decomposing Software Functions

- Including plausibility check
- Comparison with other wheel speeds
- Replacement of value

\[ s_{ij} = *, \Delta s_{ij} = * \]
\[ S_{kl} = *, \Delta s_{kl} = * \]
\[ \Delta s_{ij} = 0 \]
FMEA of Cyber-physical Systems
Interacting and Communicating Systems
Interaction of Subsystems

- Propagation of $\Delta$s
- Late/missing signals
- E.g. engine ECU: amount of injected fuel too high …
- … due to electrical short in left mirror …
Summary and Discussion
Integrated Analysis - Dichotomy

Physical components:
- Model: actual behavior
- Determined by physics
- Finite space of faults

Software:
- Model: desired behavior
- Determined by design
- Design faults
- Implementation faults …
- … infinite …
FMEA of Embedded Software

- Note:
  - Not testing!
  - Not debugging!
- FMEA: (worst-case) **qualitative** analysis of impact on overall behavior of physical system

- Well-defined **interface** software – physical system:
  - **Sensor** signals, **actuator** commands (I → O)
- Numerical distinctions → qualitative abstraction
- Metric time → qualitative abstraction
- Abstract, qualitative functional model
- Including finite set of faults
Key Issue: Modeling

- Requirement: cheap modeling
- \(\rightarrow\) compositional modeling
- \(\rightarrow\) libraries with re-usable model fragments
- Feasible?

- ValveControl\_ij: instance of generic function: signal/state change based on threshold
- Possible due to level of abstraction

- How to obtain models?
- Used as building blocks in analysis/spec.
- Automatic abstraction from requirements
- Automatic abstraction from code