53rd. IFIP WG 10.4 Meeting

Experimental Risk Assessment & Component-based Software Certification

Regina Moraes
Component-based Systems

What is the reason for using this approach?

Reuse

What is the cost if a faulty component is retrieved from the repository?

Reuse is discouraged
Component-based Systems

What may happen if we use a well tested component in a new system?

New Problems can appear

- Lack of detailed information
- Interoperability problems
- Different operational conditions

Reused Component represents a Risk to the New System
Modern Software

Lots of Software components
Different sizes
Different levels of granularity

Software Products Certification is more crucial than ever
Certification

Certified Components

Key Precondition for CBSE to be successfully adopted in the large
Key Idea

Software Certification

To guarantee our products conform to users' needs and well-defined standards:

Estimating the RISK of using a component in a larger system:

\[ \text{Risk} = \text{prob}(f) \times \text{cost}(f) \]

[Rosenberg 2000]
ISO/IEC 9126

- Quality model that focuses on software product

- **Static Properties**
- **Internal Quality**
- **External Quality**
- **Outside view of sw behavior**

- **Quality in Use**
  - Using in a particular task and environment

- **ISO 9126**
  - Functionality
  - Portability
  - Maintainability
  - Efficiency
  - Reliability
  - Usability

- **Quality in Use**
  - Effectiveness
  - Productivity
  - Safety
  - Satisfaction
Certification for Reuse – ISO/IEC 9126

Satisfaction
Satisfying the user in a specific context of use

Effectiveness
Let users reach specified targets with accuracy and completeness in a specific context of use

Safety
Present acceptable levels of risk of damage to individuals, businesses, software, property or the environment in a specific context of use

Productivity
Let users employing appropriate amount of resources in relation to the effectiveness achieved in a specific context of use
ISO/IEC 14598

- Guides the planning and the execution of an evaluation process of software quality products.
- Can be used in conjunction with ISO/IEC 9126.
- Fundamental characteristics expected in the software products evaluation process:
  - repeatability
  - reproducibility
  - impartiality
  - objectivity

We got this all with our approach.
Software Risk

How to estimate the risk on the use of components in my system?

Injection of SW faults

Risk depends on the probability of the existence of residual fault in the component
Risk depends on the residual fault activation and the impact in the system if it occurs
Software Risk

Risk_c = \( \text{prob}(f_c) \times \text{cost}(f_c) \)

Now we have a repeatable, reproducible and objective evaluation.
Experimental Risk Assessment

1. Estimate the $\text{prob}(f_c)$ by using complexity metrics of the target component in a logistic regression analysis.

2. Evaluate $\text{cost}(f_c)$ experimentally through the injection of software faults in the target component and measure its impact on the system under analysis.

3. Use a real workload and operational profile during the fault injection experiments.

4. Use a realistic distribution of faults to be injected.
Residual Fault Estimation

- Based on logistic regression that is useful to address the relationship between metrics and the fault-proneness of components

- Logistic regression equation after a linear transformation

\[
\text{logit}(\text{prob}) = \ln\left(\frac{\text{prob}}{1 - \text{prob}}\right) = \alpha + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n
\]

- Large Components
  - Consider the weight of each sub-component related to the metric that best represent the system characteristics

\[
\text{prob}_g(f) = \sum \text{prob}_i(f) \times (\text{Metrics}_i / \sum \text{Metrics}_i)
\]
G-SWFIT

Injection of Sw Faults based on a set of fault injection operators resulted from a field study using G-SWFIT technique

<table>
<thead>
<tr>
<th>Fault types</th>
<th>Description</th>
<th>% of total observed</th>
<th>ODC classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>MIFS</td>
<td>Missing &quot;if( cond ) { statement(s) }&quot;</td>
<td>9.96 %</td>
<td>Algorithm</td>
</tr>
<tr>
<td>MFC</td>
<td>Missing function call</td>
<td>8.64 %</td>
<td>Algorithm</td>
</tr>
<tr>
<td>MLAC</td>
<td>Missing &quot;AND EXP&quot; in expression used as branch condition</td>
<td>7.88 %</td>
<td>Checking</td>
</tr>
<tr>
<td>MIA</td>
<td>Missing &quot;if( cond )&quot; surrounding statement(s)</td>
<td>4.32 %</td>
<td>Checking</td>
</tr>
<tr>
<td>MLPC</td>
<td>Missing small and localized part of the algorithm</td>
<td>3.19 %</td>
<td>Algorithm</td>
</tr>
<tr>
<td>MVAE</td>
<td>Missing variable assignment using an expression</td>
<td>3.00 %</td>
<td>Assignment</td>
</tr>
<tr>
<td>WLEC</td>
<td>Wrong logical expression used as branch condition</td>
<td>3.00 %</td>
<td>Checking</td>
</tr>
<tr>
<td>WVAV</td>
<td>Wrong value assigned to a value</td>
<td>2.44 %</td>
<td>Assignment</td>
</tr>
<tr>
<td>MVI</td>
<td>Missing variable initialization</td>
<td>2.25 %</td>
<td>Assignment</td>
</tr>
<tr>
<td>MVAV</td>
<td>Missing variable assignment using a value</td>
<td>2.25 %</td>
<td>Assignment</td>
</tr>
<tr>
<td>WAEF</td>
<td>Wrong arithmetic expression used in parameter of function call</td>
<td>2.25 %</td>
<td>Interface</td>
</tr>
<tr>
<td>WPV</td>
<td>Wrong variable used in parameter of function call</td>
<td>1.50 %</td>
<td>Interface</td>
</tr>
<tr>
<td>Total faults coverage</td>
<td></td>
<td>50.69 %</td>
<td></td>
</tr>
</tbody>
</table>
Distribution of Fault Injected

- The distribution of the number of faults to inject in each component is based on its fault proneness estimation through logistic regression.
- For large components with a very large number of fault locations, faults are internally distributed according to the distribution observed in field study.
- For small components with a small number of fault locations, faults are distributed using the best approximation of the distribution observed in field study.
Evaluation of the Cost

- After the injection of each fault, the cost is measured as the impact observed in the whole system as a consequence of the fault injected in the component.

- The results measured by using fault injection include the probability of fault activation and the consequence of a failure, both measured through the impact observed.

\[ \text{cost}(f) = \text{prob}(fa) \times c(\text{failure}) \]
**Failure Modes**

- **Hang** – when the application is not able to terminate in the pre-determined time
- **Crash** – the application terminates abruptly before the workload is completed
- **Wrong** – the workload terminates but the results are not correct
- **Correct** – there are no errors reported and the result is correct
The Case Study

Satellite Data Handling Software (ESA)

Software fault Injection (G-SWFIT)

DHS

PR

PL

RTEMS or RTLinux

CDMS

Ground Control

Linux

Commands

Telemetry

RS232

The Case Study

Satellite Data Handling Software (ESA)
## Results - Metrics & Coefficients

### Fault Density Likelihood Estimation

<table>
<thead>
<tr>
<th>Metrics</th>
<th>RTLinux</th>
<th>Coefficients</th>
<th>p-value</th>
<th>RTEMS</th>
<th>Coefficients</th>
<th>p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>C. Complexity</td>
<td>39604</td>
<td>0.0072393</td>
<td>6.51 E-11</td>
<td>28536</td>
<td>0.0063537</td>
<td>7.09 E-05</td>
</tr>
<tr>
<td>N. Parameters</td>
<td>10778</td>
<td>-0.0051718</td>
<td>0.185622</td>
<td>8454</td>
<td>0.0117627</td>
<td>0.012413</td>
</tr>
<tr>
<td>N. Returns</td>
<td>13268</td>
<td>0.0431363</td>
<td>1.75 E-52</td>
<td>10240</td>
<td>0.0161907</td>
<td>0.000616</td>
</tr>
<tr>
<td>Progr. Length</td>
<td>1172521</td>
<td>-0.0001692</td>
<td>0.001896</td>
<td>787949</td>
<td>-0.0005537</td>
<td>7.9 E-20</td>
</tr>
<tr>
<td>Vocab. Size</td>
<td>171408</td>
<td>0.0011511</td>
<td>3.69 E-05</td>
<td>108550</td>
<td>0.0104020</td>
<td>2.48 E-47</td>
</tr>
<tr>
<td>Max. Nest. Depth</td>
<td>3963</td>
<td>0.3746203</td>
<td>1.0 E-140</td>
<td>2478</td>
<td>0.2354918</td>
<td>3.88 E-27</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Application</th>
<th># Module</th>
<th>LoC</th>
<th>C. Complexity</th>
<th>Global</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>&lt; 100</td>
<td>100- 400</td>
</tr>
<tr>
<td>RTEMS</td>
<td>1257</td>
<td>87,0%</td>
<td>11,0%</td>
<td>2,0%</td>
</tr>
<tr>
<td>RTLinux</td>
<td>2212</td>
<td>90,0%</td>
<td>9,0%</td>
<td>1,0%</td>
</tr>
</tbody>
</table>
Results - Failure Modes Obtained

Cost (or Impact) Estimation

RTLinux Results

<table>
<thead>
<tr>
<th>Mode</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correct</td>
<td>50%</td>
</tr>
<tr>
<td>Wrong</td>
<td>1%</td>
</tr>
<tr>
<td>Crash</td>
<td>25%</td>
</tr>
<tr>
<td>Hang</td>
<td>24%</td>
</tr>
</tbody>
</table>

RTEMS Results

<table>
<thead>
<tr>
<th>Mode</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Correct</td>
<td>74%</td>
</tr>
<tr>
<td>Wrong</td>
<td>12%</td>
</tr>
<tr>
<td>Crash</td>
<td>9%</td>
</tr>
<tr>
<td>Hang</td>
<td>5%</td>
</tr>
</tbody>
</table>
Risk Evaluation and Certification

Risk Evaluation: \( \text{Risk}_c = \text{prob}(f_c) \times \text{cost}(f_c) \)

<table>
<thead>
<tr>
<th>Component</th>
<th>( \text{prob}(f) )</th>
<th>Cost</th>
<th>Risk</th>
<th>( \text{prob}(f) )</th>
<th>Cost</th>
<th>Risk</th>
<th>( \text{prob}(f) )</th>
<th>Cost</th>
<th>Risk</th>
<th>( \text{prob}(f) )</th>
<th>Cost</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>RTEMS</td>
<td>0.0749</td>
<td>0.09</td>
<td>0.67%</td>
<td>0.05</td>
<td>0.37%</td>
<td>0.12</td>
<td>0.89%</td>
<td>0.26</td>
<td>1.94%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RTLinux</td>
<td>0.0650</td>
<td>0.25</td>
<td>1.62%</td>
<td>0.01</td>
<td>0.06%</td>
<td>0.24</td>
<td>1.56%</td>
<td>0.50</td>
<td>3.25%</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Certification

<table>
<thead>
<tr>
<th>Component</th>
<th>( \text{prob}(f) )</th>
<th>Cost</th>
<th>Risk</th>
<th>( \text{prob}(f) )</th>
<th>Cost</th>
<th>Risk</th>
<th>( \text{prob}(f) )</th>
<th>Cost</th>
<th>Risk</th>
<th>( \text{prob}(f) )</th>
<th>Cost</th>
<th>Risk</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standard</td>
<td>*</td>
<td></td>
<td></td>
<td>*</td>
<td></td>
<td></td>
<td>*</td>
<td></td>
<td></td>
<td>*</td>
<td></td>
<td></td>
</tr>
<tr>
<td>RTEMS</td>
<td>0.0749</td>
<td>0.09</td>
<td>0.67%</td>
<td>0.05</td>
<td>0.37%</td>
<td>0.12</td>
<td>0.89%</td>
<td>0.26</td>
<td>1.94%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RTLinux</td>
<td>0.0650</td>
<td>0.25</td>
<td>1.62%</td>
<td>0.01</td>
<td>0.06%</td>
<td>0.24</td>
<td>1.56%</td>
<td>0.50</td>
<td>3.25%</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Contributions & Conclusions

This work presents a first proposal to certify a component-based system using experimental risk assessment.

Our risk equation considers the fault probability, the probability of fault activation, the probability of consequent deviation in the component behavior and the consequence of a failure.

Our approach assures a repeatable way of evaluating risk and removes the dependence on the evaluators that characterize classical risk evaluation approach.
Future Works

- To refine the risk evaluation considering other aspects in order to obtain a more realistic measure of software component risk

- To improve the certification measurement

- To define threshold value for some product line to improve certification of software system based on risk assessment
References & Works


Moraes, R., Durães, J., Martins, E., Madeira, H. “A field data study on the use of software metrics to define representative fault distribution” “Workshop on Empirical Evaluation of Dependability and Security (WEEDS) - The International Conference on Dependable Systems and Networks” – DSN 06.

Thank you for your attention

regina@ceset.unicamp.br