53rd. IFIP WG 10.4 Meeting

Experimental Risk Assessment & Component-based Software Certification

Regina Moraes

UNICAMP

Component-based Systems

What is the reason for use this approach?

Reuse

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

Reuse is discouraged

Component-based Systems

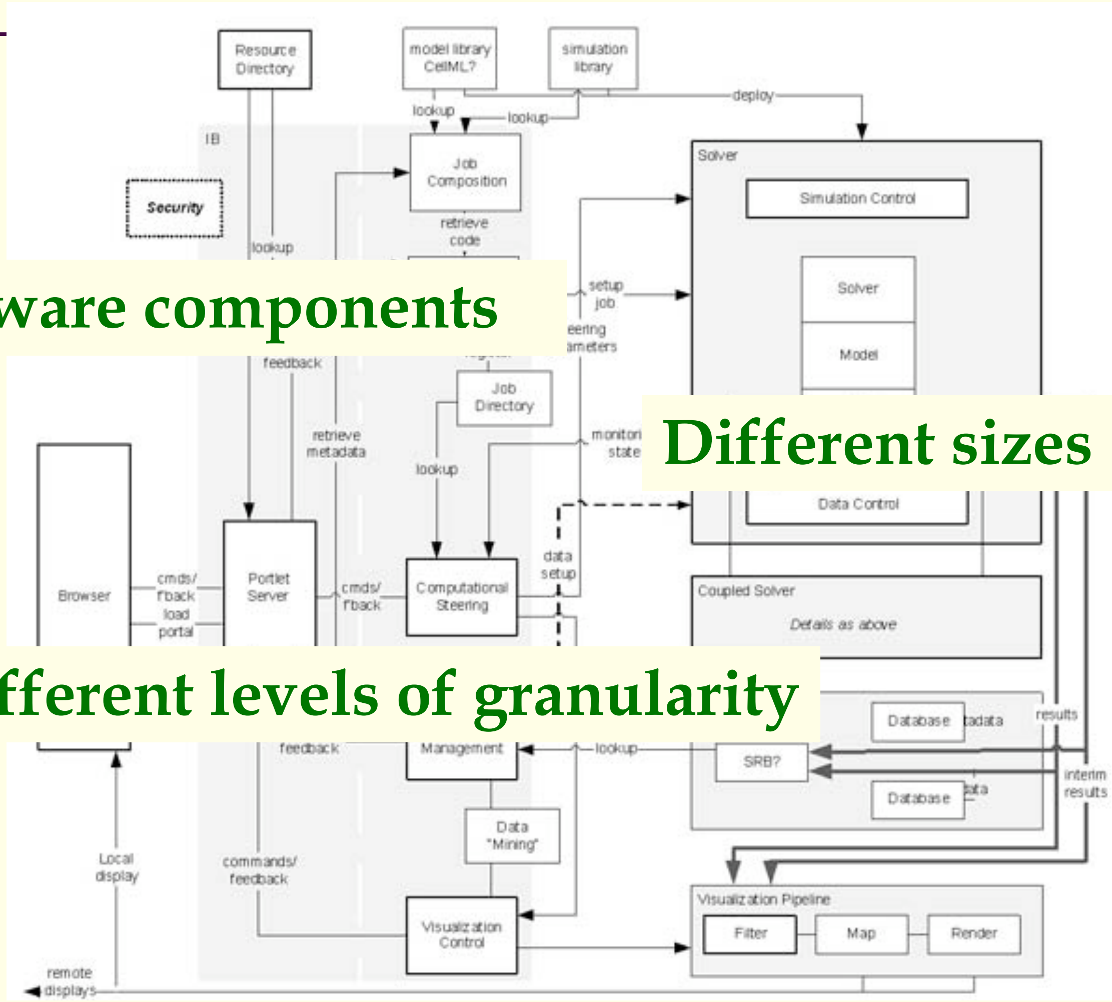
What may happen if we use a well tested component in a new system?

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 succesfully adopted in the large

Key Idea

Software Certification

Estimating the RISK of using a COMPONENT in a larger system

to Users needs and

Well-defined Standards

Risk = $prob(f) * cost(f)$

[Rosenberg 2000]

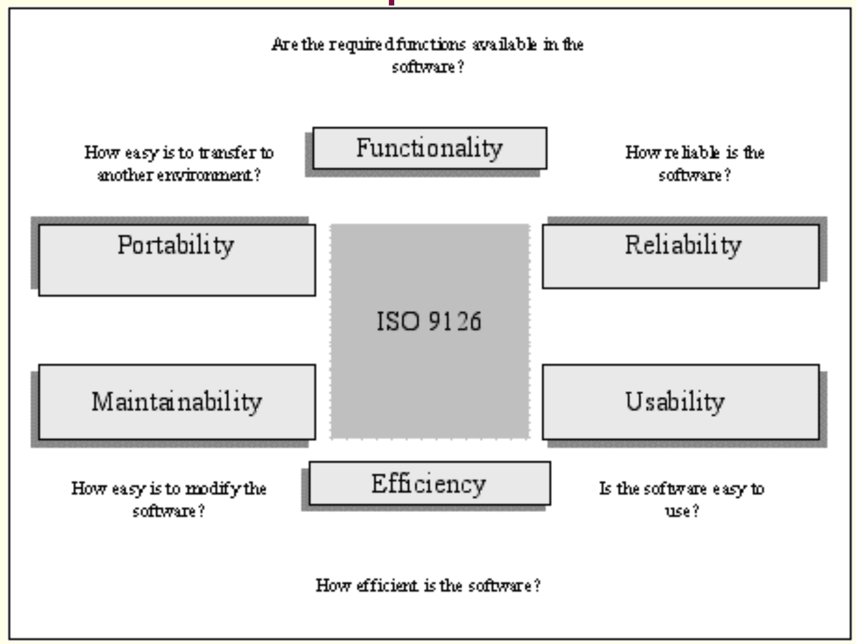
ISO/IEC 9126

- Quality model that focuses on software product

Static Properties

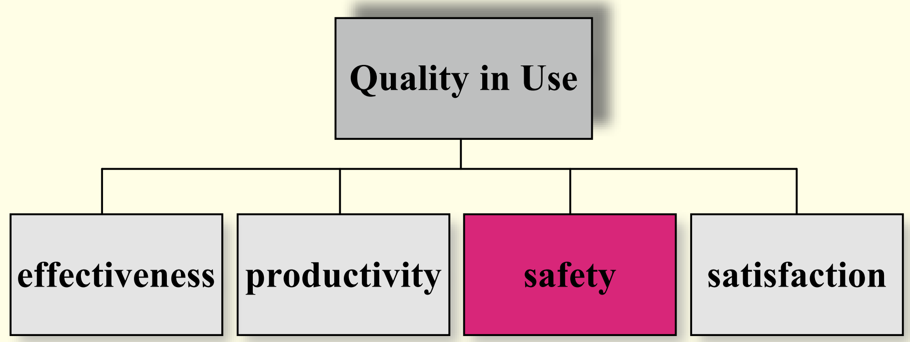
- Internal Quality
- External Quality

Outside view of sw behavior

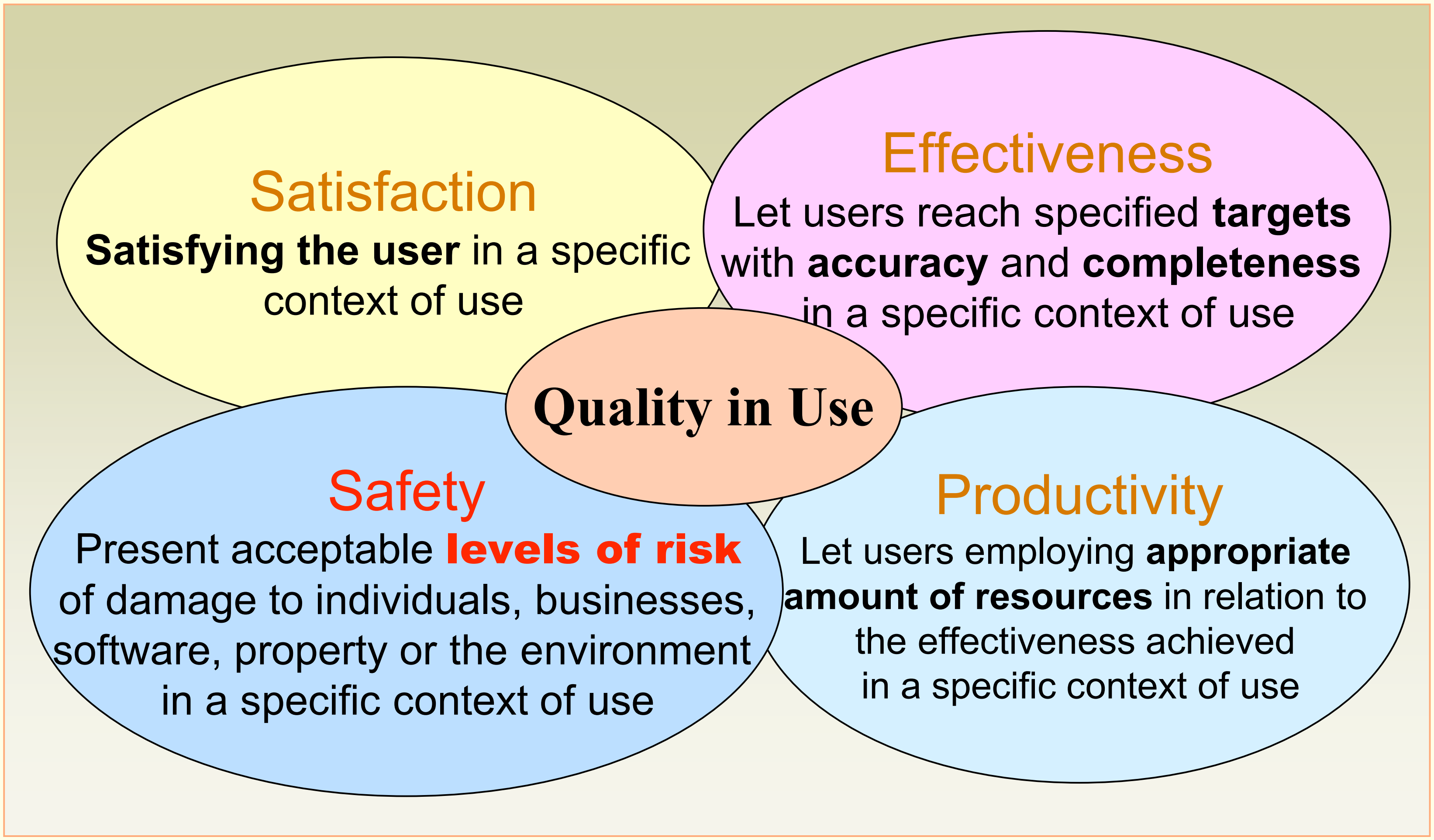


Quality in Use

Using in a particular task and environment



Certification for Reuse – ISO/IEC 9126



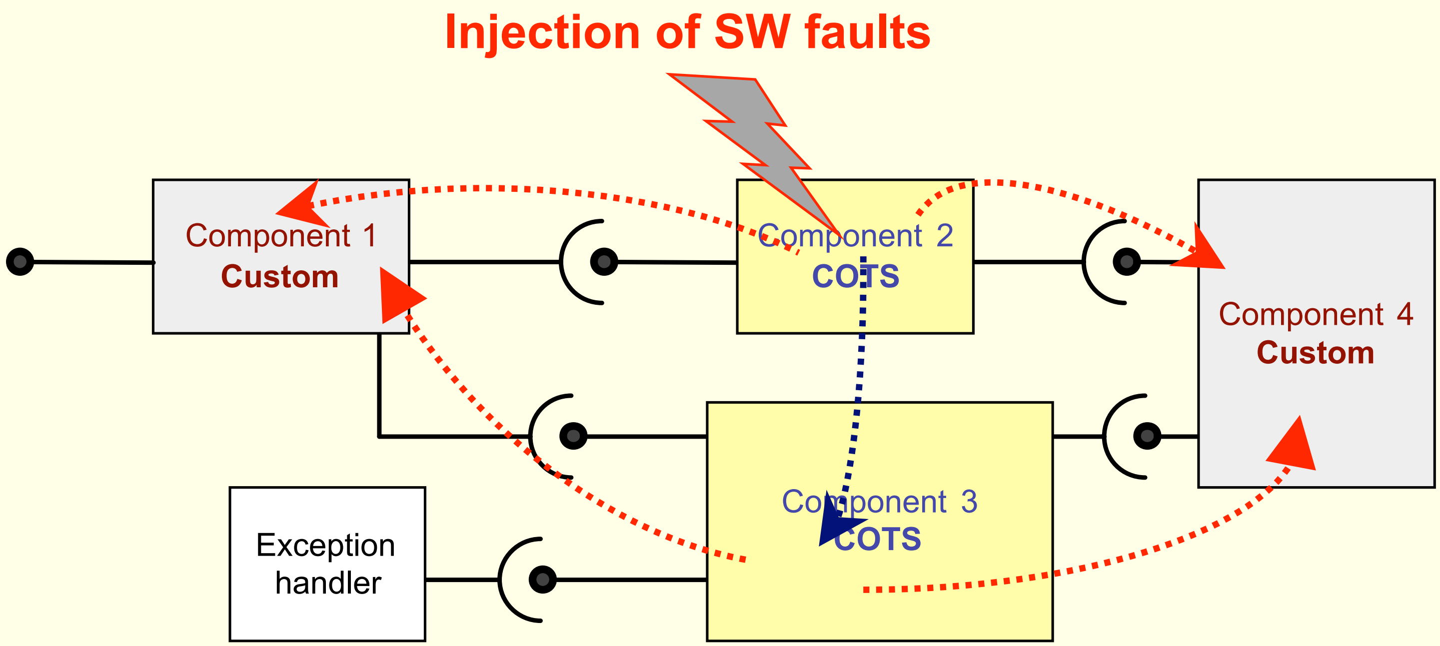
ISO/IEC 14598

- Guides the planning and the execution of a evaluation process of software quality product
- 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?



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

Statistical Model based on Logistic Regression

$$Risk_c = prob(f_c) * cost(f_c)$$

*prob(**factivated**)*c(failure)*

Software Fault Injection

Now we have a repeatable, reproducible and objective evaluation

Experimental Risk Assessment

- Estimate the *prob*(f_c) by using complexity metrics of the target component in a logistic regression analysis
- Evaluate *cost*(f_c) experimentally through the injection of software faults in the target component and measure its impact on the system under analysis
- Use a real workload and operational profile during the fault injection experiments
- 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

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

metrics

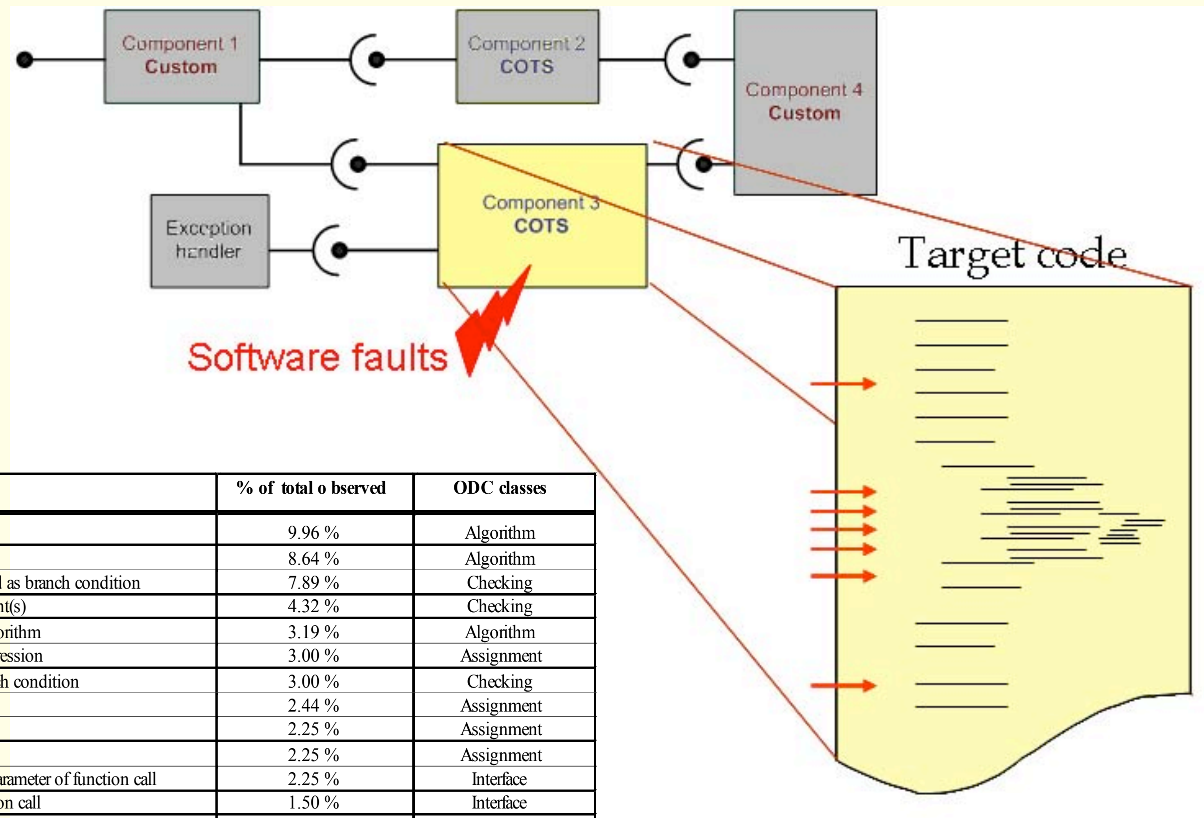
regression coefficient

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

$$prob_g(f) = \sum prob_i(f) * (Metrics_i / \sum Metrics_i)$$

G-SWFIT

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



Fault types	Description	% of total o bserved	ODC classes
MIFS	Missing "If (*cond*) { statement(s) }"	9.96 %	Algorithm
MFC	Missing function call	8.64 %	Algorithm
MLAC	Missing "AND EXPR" in expression used as branch condition	7.89 %	Checking
MIA	Missing "if (*cond*)" surrounding statement(s)	4.32 %	Checking
MLPC	Missing small and localized part of the algorithm	3.19 %	Algorithm
MVAE	Missing variable assignment using an expression	3.00 %	Assignment
WLEC	Wrong logical expression used as branch condition	3.00 %	Checking
WVAV	Wrong value assigned to a value	2.44 %	Assignment
MVI	Missing variable initialization	2.25 %	Assignment
MVAV	Missing variable assignment using a value	2.25 %	Assignment
WAEP	Wrong arithmetic expressi on used in parameter of function call	2.25 %	Interface
WPFV	Wrong variable used in parameter of function call	1.50 %	Interface
	Total faults coverage	**50.69 %**	

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

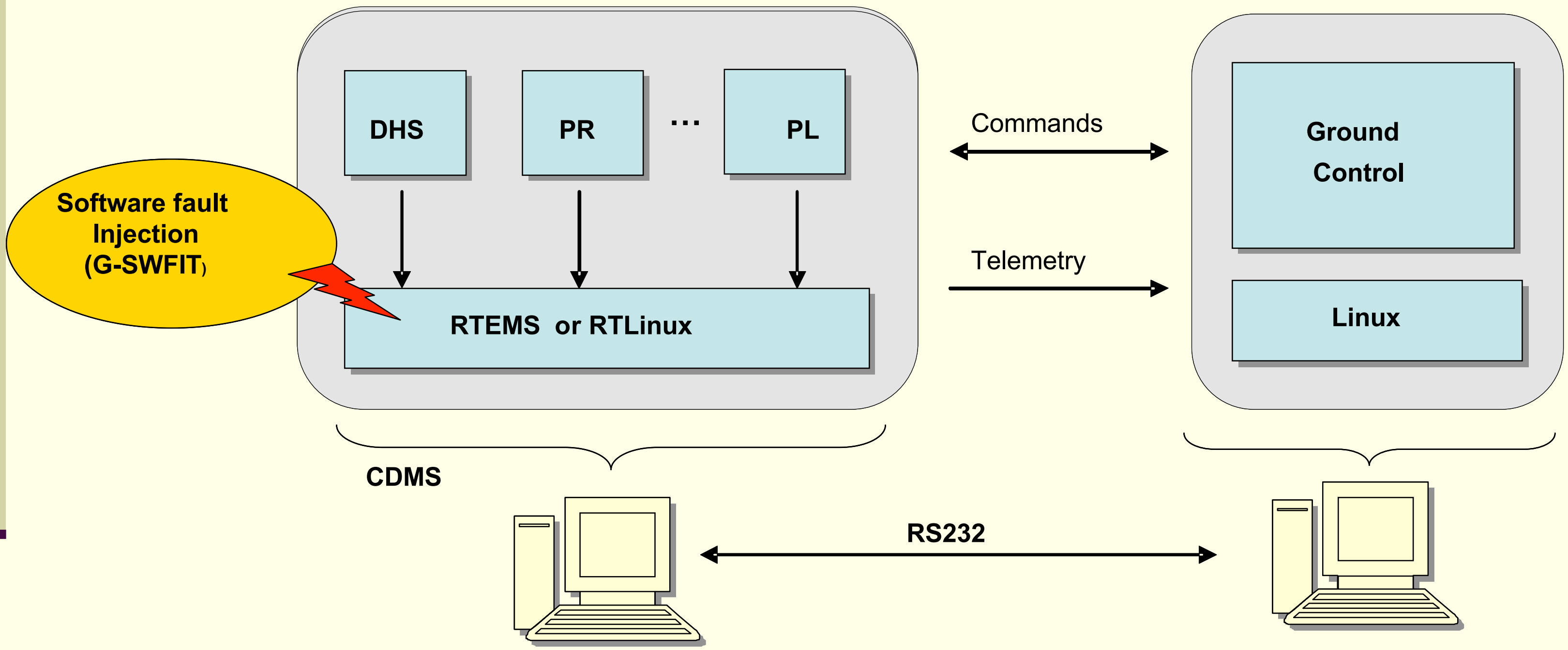
$$cost(f) = prob(fa) * \mathrm{c}(failure)$$

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)



Results - Metrics & Coefficients

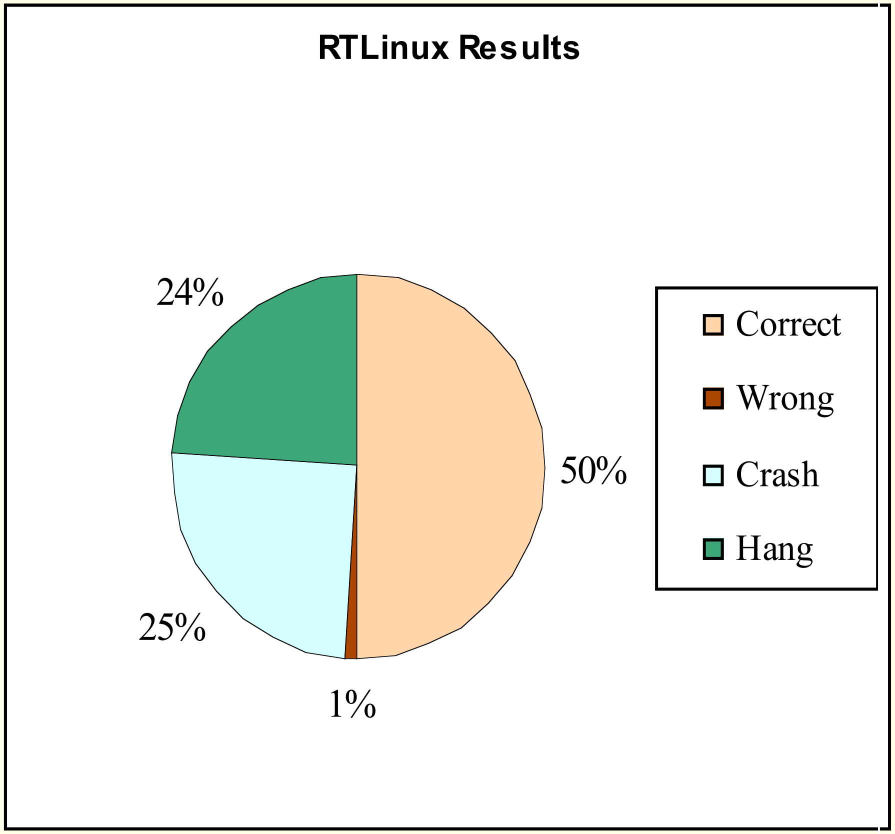
Fault Density Likelihood Estimation

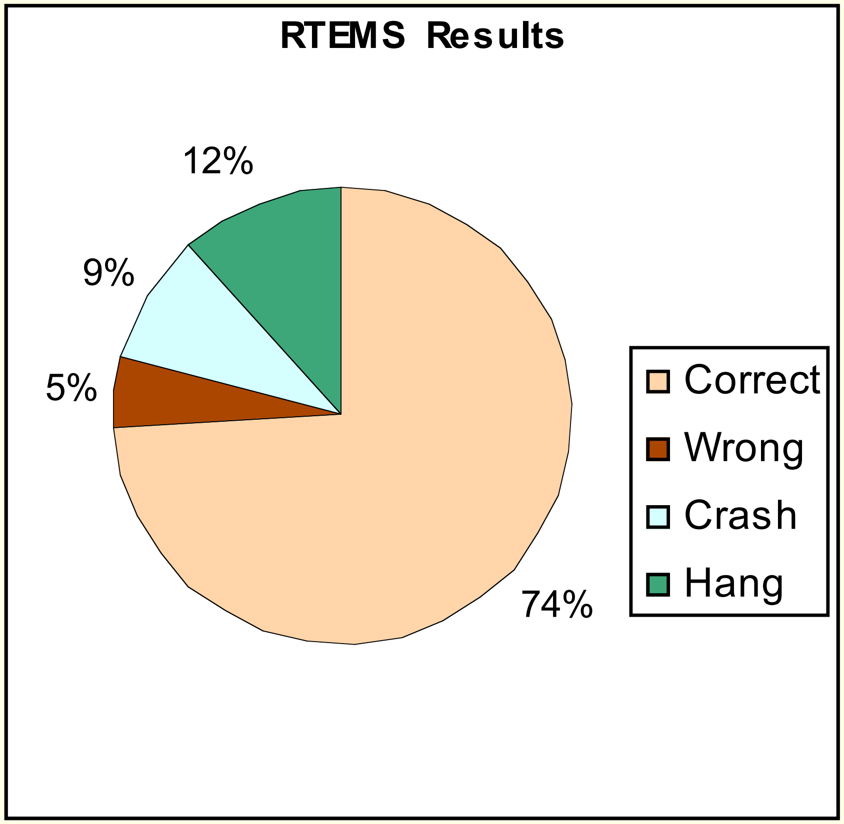
Metrics	RTLinux			RTEMS		
	Global Values	Coefficients	*p-value*	Global Values	Coefficients	*p-value*
C. Complexity	39604	0.0072393	6.51 E-11	28536	0.0063537	7.09 E-05
N. Parameters	10778	-0.0051718	0.185622	8454	0.0117627	0.012413
N. Returns	13268	0.0431363	1.75 E-52	10240	0.0161907	0.000616
Progr. Length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

Application	# Module	LoC			C. Complexity			Global
		< 100	100- 400	> 400	< 25	25-40	> 40	*prob $_g$(f)*
RTEMS	1257	87,0%	11,0%	2,0%	80,0%	6,0%	14,0%	7,5%
RTLinux	2212	90, 0%	9,0%	1,0%	84,0%	6,0%	10,0%	6,5%

Results - Failure Modes Obtained

Cost (or Impact) Estimation





Risk Evaluation and Certification

Risk Evaluation

$(Risk_c = prob(f_c) * cost(f_c))$

Component	prob(f)	Crash		Wrong		Hang		Incorrect Behavior	
		cost(f)	risk	cost(f)	risk	cost(f)	risk	cost(f)	risk
RTEMS	0.0749	0.09	0.67%	0.05	0.37%	0.12	0.89%	0.26	1.94%
RTLinux	0.0650	0.25	1.62%	0.01	0.06%	0.24	1.56%	0.50	3.25%

Certification

Component	prob(f)	Crash		Wrong		Hang		Incorrect Behavior	
		cost(f)	risk	cost(f)	risk	cost(f)	risk	cost(f)	risk
Standard			**1%**		**0.5%**		**1%**		**2%**
RTEMS	0.0749	0.09	0.67%	0.05	0.37%	0.12	0.89%	0.26	1.94%
RTLinux	0.0650	0.25	1.62%	0.01	0.06%	0.24	1.56%	0.50	3.25%

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

- Rosenberg, L., Stapko, R., Gallo, A. "Risk-based Object Oriented Testing". In: *Proc of. 13th International Software / Internet Quality Week-QW*, San Francisco, California, USA, 2000.
- ISO/IEC 9126-1. International Organization For Standardization ISO/IEC 9126-1, Software Engineering – Software product quality – Part 1: Quality Model; Geneve ISO, 2001.
- 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.*
- Moraes, R., Durães, J., Barbosa, R., Martins, E., Madeira, H. "*Experimental Risk Assessment using Software Fault Injection*", "*The International Conference on Dependable Systems and Networks*"–*DSN 07*.

Thank you for your attention

regina@ceset.unicamp.br