

# ***Workshop on Dependability Benchmarking***

39th Meeting of IFIP Working Group 10.4, Parati, Brazil, March 1-3, 2001

# *Wrap up*

- Dependability benchmarking problem space
- Session 1 - Dependability Benchmarking Approaches
- Session 2/ Session4 - Current Projects & Research
- Panels:
  - ◆ Software testing and dependability benchmarking
  - ◆ Customer point of view on dependability benchmarking
  - ◆ Fault representativeness

# *Conclusions and some thoughts - 1*

## Dependability benchmarking problem space

- ◆ The problem space is very large but we need to understand this space;
- ◆ We need to make choices to define a general framework;
- ◆ Segmentation of the application domain is needed to handle the problem;
- ◆ Critical dimensions are **measures** and **fault representativeness**.

# *Conclusions and some thoughts - 2*

## Dependability Benchmarking Approaches

- ◆ A set of dependability classes (measures) are defined for transactional servers.
- ◆ Defining a benchmark is to specify how these measures can be obtained.
- ◆ There is a market for dependability benchmarking. People/enterprises involved on dependability evaluation and validation know this.

# ***Conclusions and some thoughts - 3***

## **Current Projects & Research**

- ◆ Dbench project;
- ◆ Direct measures and models are needed;
- ◆ Unconditional dependability measures are one of the goals;
- ◆ Existing techniques/tools need to be improved;
- ◆ However, even with existing tools we can do a lot in improving existing systems. Examples from robustness testing show this quite clear.

# ***Conclusions and some thoughts - 4***

## **Panel: SW testing and dependability benchmarking**

- ◆ SW testing goal is finding faults; benchmarking doesn't care about this;
- ◆ Test sets are almost never generic!
- ◆ Standard test procedure; not a standard (unique) test set  
→ Standard benchmark framework; many benchmark
- ◆ Eliminate or standardize “non-essential” elements;
- ◆ Build a standard benchmark environment. Formalization of the benchmarks is important;
- ◆ Dependability benchmarking can also learn from SW testing:
  - Operational profiles
  - Share of SW faults in the faultload

# ***Conclusions and some thoughts - 5***

## Panel: costumer point of view on dependability benchmarking

- ◆ Application area → costumers → measures;
- ◆ One benchmark: different levels (in detail) of measures;
- ◆ Different customer classes;
- ◆ End users like simple measures;
- ◆ Simple measures are rather incomplete from a technical point of view;
- ◆ Dependability classes is a way to produce understandable measures without oversimplifying the problem;
- ◆ Classes could be useless for comparison purposes.

# ***Conclusions and some thoughts - 6***

## Panel: Fault representativeness

- ◆ Representativeness of errors is the question;
- ◆ Fault representativeness should be discussed in a statistical basis;
- ◆ The main problems are clear and are being subject of research:
  - Which classes of faults are relevant?
  - How to create a representative mix of classes of faults?
  - How to “inject” the faults in a benchmark (i.e., how to create a faultload?)?
  - How to include external features (environment, operation conditions, etc) in the faultloads?
- ◆ We can start with what we know about how to emulate faults;