

The Customer Viewpoint on Dependability Benchmarking

Neeraj Suri

Chalmers Univ.

<http://www.ce.chalmers.se/~suri>



Application scope of D.Benchmarking systems...

- “Dependability” is multi-faceted with (perceived) qualitative attributes as well – so what specific aspect(s) of dependability can a D.Benchmark (realistically/potentially) cover & measure?
 - ❖ Systems: large area covering control, telecomm., transaction systems etc → NEED to narrow down this definition
 - ❖ What are we benchmarking: Complete systems (HW, OS, middleware, applications)? Specific services (embedded)?, ...
 - ... and over user-induced changes to system functionality? (application specific or platform/service specific?)

Technical scope of D.Benchmark(s)...

- Does it cover single/distributed & networked systems and services + middleware + end-to-end system functionality?
 - Can "all" workloads, application areas, fault/stress scenarios driving the D.benchmark be covered (even to a representative extent)?
 - Is it a magic number(s)? Can it be user-trusted/reproduced? Is it a single benchmark or a suite for diff. systems/apps?
- ... How meaningful is a system benchmark, where functionality (& dependability aspects) may change based on user SW/apps.??

Who is going to use D.Benchmark(s) and for what?

Designer? for conformance to design specs/standards? [do they even exist for current day broad usage "systems"]

User? comparisons across a product class? [eg. PDA's..] utility?

- A product sells for its "*delivery of services*" to the user - if dependability is not the explicitly perceived service, then D.Benchmarks will remain as supplementary "features" to the user
- What does it "buy" the user (& designer)? Is it just a design metric/techniques/tool etc that the designer is simply expected to conform to?

Cost Issues...

- What does it cost to run benchmarks and who runs them?
 - Service or \$ critical aspects of a product need to match the cost/ease of running a benchmark ("bang for the buck" principle - *perceptions* are sometimes more valuable than a rational basis!
 - If its an embedded environment, then why should the user care if the metric is too entwined within the system ops for the user to measure, comprehend or control?

D. Benchmarking & Fault Loads etc...

- Definition of “faults” or “representative stress” needs to be carefully worked out
 - At what level should we be considering faults? OS, middleware, HW, HW-SW, application?
 - Definition of “faults” may be very system/application dependent. Eg. in RT-OS, lack of (timely) delivery of a service equates to a fault. So for benchmarking, we need to be very careful in defining what is “representative stress”!

D. Benchmarks...

- Need to communicate “value” to the user, not as a technical achievement but as a “service” commodity
- Needs to be simple & comprehensible – even a rough measure (with all the sophistication hidden) has better value for the user!
- Need to span a tangible domain (technical correctness of focused benchmarks for specific domains vs. real usage!)
- User + Application area sensitivity is “the” key dimension!!