Dependability Benchmarking for Computer Systems

Karama Kanoun, Philip Koopman, Henrique Madeira, Lisa Spainhower

Dependability Benchmarking — A Reality or a Dream?

Karama Kanoun (LAAS-CNRS, France), Phil Koopman (Carnegie Mellon University, USA), Henrique Madeira (University of Coimbra, Portugal), and Lisa Spainhower (IBM, USA)

Published in "Dependability Benchmarking for Computer Systems", Prologue, pp. xiii-xviii, IEEE Computer Society & WILEY, August 2008. ISBN: 978-0-470-23055-8

Until recently, for computer systems, a benchmark referred implicitly to performance benchmark. For example, the seminal handbook on database and transactional systems benchmarking, published in the early nineties [Jim Gray 1993], was totally dedicated to performance benchmarks. The handbook title does not even mention "performance" benchmark. De facto, a benchmark has been a performance benchmark.

Many factors converge to raise the importance of dependability of today's computer-based systems. These include global 24x7 operations, on-line businesses, and the increasing complexity of systems. At the same time, high levels of performance are available at rapidly decreasing cost, decreasing the predominance of absolute performance and relative price/performance as indicators of customer satisfaction. Dependability is more and more playing a determinant role. However, while industry standard benchmarks are easily and widely used to measure computer performance in a deterministic and reproducible manner, dependability benchmarking is in its infancy. Nonetheless, although far from the maturity of current performance benchmarks, dependability

benchmarking is making definite progress, as demonstrated by efforts in industry and academia illustrated in this book.

A dependability benchmark is intended to characterize the system behavior in the presence of faults that could be internal or external to the system being benchmarked. Potential faults include component failures, hardware or software design flaws, faults in other systems interacting with the benchmarked systems, operator errors, and perturbations in the environment. Benchmarking the dependability of a system consists of evaluating dependability or dependability-and-performance-related measures in the presence of faults, in a well-structured and standardized way. Measures may characterize the system in a comprehensive way; that is, they may address the service delivery level and take into account the occurrence of various events impacting its behavior and their consequences. On the other hand, they may characterize specific features of the system such as coverage provided by fault tolerance mechanisms, time to restart the system, or time to system backup. As a consequence, numerous benchmarking measures are of interest.

The key aspect that distinguishes benchmarking from existing evaluation and validation techniques is that a benchmark fundamentally represents an agreement (explicit or tacit) that is accepted by those who make and sell computers and those who purchase them. This technical agreement states the measures, the way the measures are obtained, and the domain (e.g., application area) in which these measures are considered valid and meaningful. In other words, a real benchmark is something that the user community and the computer industry accept as representative enough of a given application domain to be deemed useful, and to be generally used as a way of measuring specific features of a computer system and, consequently, a way to compare different systems.

Currently, several organizations and research groups are carrying out promising work on dependability benchmarking, and, as a result, a great variety of benchmarks has been defined and implemented in the last decade. Many of these dependability benchmarks resulted from work performed by single institutions (proprietary benchmarks defined in response to internal needs, or work on particular aspects of dependability benchmarking). A few other benchmarks resulted from coordinated and concerted work between several institutions. Even though standard, well established and widely agreed on dependability benchmarks, approved

by recognized consortiums as in the case of performance benchmarks, do not really exist, the dependability benchmarks developed so far are paving the way for such standard benchmarks.

The results of a dependability benchmark are aimed at either characterizing system dependability capabilities in a qualitative manner (e.g., on the basis of the dependability features supported or claimed, such as on-line error detection, fail-silent failure mode), or quantitatively assessing these properties. Dependability benchmark results can be useful for both the end-users and vendors to:

- Characterize the dependability of a component or a system, qualitatively or quantitatively.
- Track dependability evolution for successive versions of a product.
- Identify weak parts of a system, requiring more attention and perhaps needing some improvements by tuning a component to enhance its dependability, or by tuning the system architecture (e.g., adding fault tolerance) to ensure a suitable dependability level.
- Compare the dependability of alternative or competitive solutions according to one or several dependability attributes.

A dependability benchmark can be performed in various phases of the system life cycle. The measures obtained for a specific phase are then helpful for the current or subsequent phases. The purpose of a benchmark may vary significantly along the system life cycle. For example:

- During the very early design phases, a dependability benchmark could support the decision whether to purchase a particular hardware or software component or platform that is to be integrated into a particular application.
- During development, results could be used to reveal weak points and to monitor the improvement actually achieved by fault removal activities (e.g., regression testing).
- For operational life, benchmark results could be useful to evaluate the impact of faults (hardware, software or operator faults) on system dependability.

With many current systems, the traditional role of dependability assessment methods in the development life cycle of computing systems and applications (i.e., measuring used as an inherent step of the improvement

process, and as a verification and validation facilitator) must be expanded in order to address the technical problems resulting from current component-based development practices involving intensive reuse of components. The use of dependability benchmarks is an important step in that direction. In fact, it is now a common practice for large scale software development to reuse pre-existing ("off-the-shelf") components (normally, general-purpose components possibly from the open source community or components easily adapted to the requirements of the new system) and to develop from scratch only such further components and “glue” code as are also needed. Given the high costs of designing and implementing new software, vendors see the reuse of components as a way to reduce development effort and to achieve short rapid time-to-market. However, component reuse introduces unknown risks of failure, as the new operational conditions may differ substantially from those that the components were initially designed for, and the new operational conditions may cause the activation of unknown residual faults or produce new component interaction faults. The use of dependability benchmark in the development/integration of such composite systems seems very useful for component selection and to assess dependability measures of the whole system.

The benchmark performer (i.e., the person or entity actually performing the benchmark) can be a system manufacturer (or vendor), a system integrator, a third party, or an end-user. These entities have different visions of the target system and, as a consequence, they have diverse expectations from the benchmark results. For example, system vendors have low-level access and observation points, while other entities usually can only make use of available input and outputs to interact with the system and observe its behavior to assess its dependability.

To sum up, dependability benchmarks allow objective characterization of system dependability. Therefore, they can provide a good means for fair comparison between alternative systems. They can also be used for guiding development efforts of system providers, and for supporting acquisition choices of system purchasers, or for comparing the dependability of new versions of a system with respect to previous ones.

Due to the above various aspects of dependability benchmarks, a range of approaches for dependability benchmarking has been followed. This book reflects these varieties. Some of them address the end-user points of

views and others address primarily the vendor point of view (even though the user point of view is always there). Additionally, these benchmarks are at different maturity stages.

Benchmarking a system or a component is typically based on experimentation, or modeling, or on both. Experimental results may be obtained from controlled experiments defined in the benchmark or from the observation of the system in the field, under its operational conditions. Controlled experimentation consists of applying a workload and a faultload to the system under benchmark to obtain the benchmark measures; they are based on fault injection techniques. All the benchmarks presented in this book can be understood in terms of the reference model for implementing dependability benchmarks illustrated in Figure 1. However, some work focuses mainly on the controlled experimentation parts (putting effort into the selection of the workload, faultload and experimental measures), some work gives more importance to modeling, while other work emphasizes obtaining experimental results from system observation.

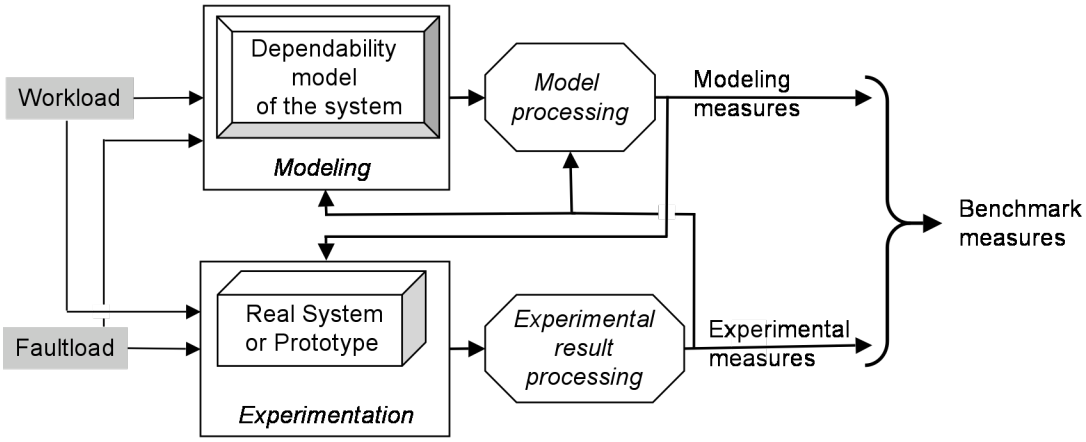

Figure 1: **Reference model for implementing dependability benchmarks**

In general, it is clear that dependability benchmarking is still a developing art. It took decades for performance benchmarking to mature, progressing from measures such as Whetstones, Dhrystones, and Livermore Loops to multiple generations of SPEC benchmarks. Primary technical challenges were in the areas of representativeness of the workloads, scalability of benchmarks to larger systems, and vulnerability of benchmarks to designing and tuning systems to optimize performance of particular benchmarks aggressive compiler optimizations. It should,

therefore, be reasonable to expect that dependability benchmarking will similarly take many years to reach maturity.

There are some aspects of dependability benchmarking that are likely to prove even more challenging than achieving well accepted performance benchmarks. First, actual performance is something that can be measured in a reasonably short amount of time (hours or days) in an actual production system. Thus, the accuracy of a performance prediction can be validated fairly quickly with respect to a particular user's everyday experience. On the other hand, measuring achieved dependability requires operating for a long enough period of time to observe how the system responds to real failure. Given a reasonably reliable system, it could take months or years to find out if a dependability benchmarking prediction bears any resemblance to actual experience for a given system.

An additional complication with creating a good dependability benchmark is the public unavailability of comprehensive fault and failure data from which to create a faultload. Addressing this lack will at the very least be analogous to the process by which performance benchmarking gathered representative user programs through the creation of the SPEC (Standard Performance Evaluation Corporation) consortium. However, collecting faultloads is likely to be more challenging because many faults depend heavily on implementation and operating environment (they are not portable in the way source programs are). Furthermore, many of the best sources of such data have justifiable competitive disincentives to make data available on what has gone wrong with their systems.

Current trends in computing system design contribute to making dependability benchmarking increasingly difficult. For instance, the highly dynamic nature of most systems and the constant adaptation to changes in the environment, particularly in large networked systems composed of heterogeneous nodes that demand online deployment of services, runtime reconfiguration and upgrading, make the benchmarking of dependability attributes a difficult, but extremely important technical challenge. But, the change from single computing nodes to networked systems of systems has also been a challenge for traditional performance benchmarking. Like performance benchmarks, dependability benchmarks will need to continually evolve to keep pace with the changing nature of the computing world.

We expect that eventually there will be a solid set of accepted, standardized, and validated dependability benchmarks. As with performance benchmarking success will be as much of a journey of continually improving benchmarks to keep up with technological changes. But, the rewards of such a journey will be plentiful.

The idea of preparing this book arose during the completion of DBench, a European project on Dependability Benchmarking, partially supported by the European Commission during three years, 2000-2004 [DBench], and has matured and evolved within the Special Interest Group on Dependability Benchmarking [SIGDeB], founded in 1999 by the IFIP Working Group 10.4 on Dependable Computing and Fault Tolerance.

DBench developed a framework for defining dependability benchmarks for computer systems, with emphasis on *Off-the-Shelf components*, (commercial or not) and on systems based on Off-the-Shelf components, via experimentation and modelling [Kanoun et al. 2001]. To exemplify how the benchmarking issues can actually be handled in different application domains, a set of benchmarks and their associated implementations has been developed. They concern general-purpose operating systems, embedded systems (automotive and space applications), and transactional systems. These benchmarks are presented in this book. They address specifically the end-user and the system integrator points of view. They share the following common characteristics:

- The benchmark performer is the system purchaser or a third party who has no in depth knowledge about the benchmark target and who is aiming at obtaining valuable information about the target system dependability.
- The primary users of the benchmark results are the end-users of the benchmark target or the *integrators* of the system including the benchmark target.

On the other hand, the work of the Special Interest Group on Dependability Benchmarking started with an open-ended mission of exploration [Koopman & Madeira 1999], but evolved to consider a system vendor point of view [Wilson et al., 2002]. The benchmark performer, who is the system developer, has access to detailed information on the system that is not available for system purchasers. The work has identified a set of standardized classes for characterizing the dependability of computer systems. The classification seeks to enable comparison of different computer systems in the dimensions of availability, data integrity, disaster recovery, and

security. Different sets of criteria have been proposed for computer systems that are used for different application types (e.g. transaction processing and process control).

Pioneer work on dependability benchmarking is published in [Tsai et al. 1996] for fault tolerant systems, in [Mukherjee & Siewiorek 1997] for software systems, and in [Brown and Patterson 2000] for Software RAID systems. The first open workshop on the topic was held by the Special Interest Group on Dependability Benchmarking in 2002 [Workshop 2002]. Much of the work on dependability benchmarking has been published after 2000, authored primarily by institutions and individuals contributing to this book.

References

[Brown and Patterson 2000] A. Brown and D.A. Patterson, "Towards Availability Benchmarks: A Case Study of Software RAID Systems", in Proc. 2000 USENIX Annual Technical Conference, San Diego, CA, USA, USENIX Association, 2000.

[DBench] http://www.laas.fr/DBench

[Gray 1993] Jim Gray (Editor), The Benchmark Handbook for Database and Transaction Systems (2nd Edition), Morgan Kaufmann, 1993, ISBN 1-55860-292-5

[Kanoun et al. 2001] K. Kanoun, J. Arlat, D. Costa, M. Dalcin, P. Gil, J.-C. Laprie, H. Madeira, N. Suri, "DBench – Dependability Benchmarking", Supplement of the Int. Conf. on Dependable Systems and Networks, Göteborg, Sweden, 2001, pp. D.12-D.15.

[Koopman & Madeira 1999] P. Koopman and H. Madeira, Dependability Benchmarking & Prediction: A Grand Challenge Technology Problem", *1st International Workshop on Real-Time Mission-Critical Systems: Grand Challenge Problems*, November 1999, 4 pages, Phoenix, AZ, USA.

[Mukherjee & Siewiorek 1997] A. Mukherjee and D. P. Siewiorek, "Measuring Software Dependability by Robustness Benchmarking", *IEEE Transactions of Software Engineering*, vol. 23 no. 6, pp. 366-376, 1997.

[SIGDeB] http://www.laas.fr/~kanoun/ifip_wg_10_4_sigdeb/

[Tsai *et al*. 1996] T. K. Tsai, R. K. Iyer and D. Jewitt, "An Approach Towards Benchmarking of Fault-Tolerant Commercial Systems", in *Proc. 26th Int. Symp. on Fault-Tolerant Computing (FTCS-26),* (Sendai, Japan), pp. 314-323, IEEE CS Press, 1996.

[Wilson et al. 2002] D. Wilson, B. Murphy and L. Spainhower, "Progress on Defining Standardized Classes for Comparing the Dependability of Computer Systems," *Workshop on Dependability Benchmarking,* pp. F1-5, Washington, D.C., USA, 2002.

[Workshop 2002] Workshop on Dependability Benchmarking, Supplement Volume of 2002 *International Conference on Dependable Systems and Networks (DSN)*, July 2002, pp. F1-F36, IEEE CS press. Also, papers are available at: http://www.laas.fr/~kanoun/ifip_wg_10_4_sigdeb/external/02-06-25/index.html.