Editor in Chief: M. Satyanarayanan ■ Carnegie Mellon University ■ satya@cs.cmu.edu

Metrics and Benchmarks for Pervasive Computing

M. Satyanarayanan

When you can measure what you are speaking about, and express it in numbers, you know something about it; but when you cannot measure it, when you cannot express it in numbers, your knowledge is of a meager and unsatisfactory kind; it may be the beginning of knowledge, but you have scarcely in your thoughts advanced to the state of Science, whatever the matter may be.
—Lord Kelvin, 1883

Over a long and distinguished career, Lord Kelvin (1824–1907) made fundamental contributions to many areas of science and technology. Our modern unit of temperature on the absolute scale (degrees Kelvin) is named after him. His theoretical work on submarine telegraphy enabled the first commercially successful transatlantic telegraph system in 1865. This reduced roundtrip communication latency between the United States and Britain by six orders of magnitude: from weeks to seconds.

The quote above reflects Kelvin's enduring belief in the importance of quantifying knowledge that is qualitative and hence fuzzy. I have found his message to be true in my own research—the effort to quantify and measure a phenomenon is rewarded by a deeper understanding of that phenomenon and by rich insights into its implications.

QUANTIFYING THE BENEFITS

As pervasive computing emerges from infancy, it would do well to heed Kelvin's advice. We can accelerate the growth and success of the field by creating metrics and benchmarks that quantify benefits and costs at scales ranging from individual algorithms to complete systems (including users).

In addition to Kelvin's scientific motivation, metrics and benchmarks can also help in many pragmatic ways. First, they make progress visible. Pervasive computing, by its very nature, is about making computing invisible. How do you demonstrate progress in invisibility? The better your system, the less there is to see! Only through detailed internal and external measurements can innovation be made visible.

Second, by helping to define intermediate milestones toward a distant goal, metrics and benchmarks can help sustain long-term research. The funding for such research in both academia and industry critically depends on demonstrating incremental progress. Metrics and benchmarks will help us create a roadmap against which progress can be calibrated and the need for corrections detected in a timely manner.

Third, metrics and benchmarks stimulate competition by defining a framework for comparing the effectiveness of alternative approaches. For example, in database research, the TPC family of benchmarks has helped to drive progress in industry and academia (see www.tpc.org/information/benchmarks.asp). Similarly, in computer architecture research, the SPEC family of benchmarks has provided a common ground upon which the merits of alternative designs can be compared (see www.spec.org/benchmarks.html).

RECOGNIZING THE CHALLENGES

We face at least three challenges in trying to quantify pervasive computing. The first is *combining realism with reproducibility*. Typical pervasive computing scenarios involve many unpredictable components such as human actions and reactions, mobility patterns, and variation in computing resources such as wireless-network bandwidth and battery level. It is not easy to define a benchmark in a way that preserves realism yet gives the same results for identical experiments. Rigid control improves reproducibility but hurts realism. What is the point of a benchmark that has little relevance to how a system

MISSION STATEMENT: *IEEE Pervasive Computing* is a catalyst for advancing research and practice in mobile and ubiquitous computing. It is the premier publishing forum for peer-reviewed articles, industry news, surveys, and tutorials for a broad, multidisciplinary community.

is used in practice? Striking the right balance is an art rather than a science.

One promising technique is to obtain realism by capturing traces of the unpredictable components and to obtain reproducibility by replaying those traces. For example, wireless-network quality[1] and distributed file system performance[2] have been successfully investigated using trace replay. Perhaps we can use trace replay more broadly in pervasive computing.

A second challenge is the *sheer complexity of pervasive computing systems*. This makes it difficult to interpret measurements and determine the contribution of different system components to those measurements. It also makes it difficult to identify bottlenecks on which attention should be focused to improve the whole system. Perhaps we can adapt the approaches of profiling tools, such as gprof (for CPU cycles[3]) and PowerScope (for energy[4]) for other types of measurements in pervasive computing. These tools periodically interrupt a system while it is running to record critical elements of its state. Careful design and implementation can keep the perturbation caused by these periodic interrupts low. Later, in postprocessing, more disruptive analyses can be performed. Sunny Consolvo and Miriam Walker's Experience Sampling Method can be viewed as an application of this technique to user-level activities.[5]

A third challenge is the *multidisciplinary nature of pervasive computing*. An end-to-end quantitative analysis of a pervasive computing system will need to integrate the analyses of user behavior, software behavior, hardware behavior, wireless-network behavior, and so on. Today, these fall under the purviews of distinct research communities, each with their own norms of experimental methodology. The only way to bridge this wide diversity is for members of each community to broaden their perspective and to learn the mores of the others. Such broadening is, of course, a goal of this publication.

HARDENING SOFT ATTRIBUTES

Cost is often easier to quantify than benefit. For example, measuring the CPU, memory, network bandwidth, and energy overheads of a software component intended to reduce user distraction is relatively straightforward. These represent the costs of using the software component. But how do we measure user distraction?

Without a suitable metric, quantifying the benefit of using the software component is impossible. A good metric must have excellent correlation with the attribute of interest and must be easy to measure. Temperature, for example, correlates well with our feeling of warmth or cold; it is easily measured by the length of a liquid column in a glass tube called a thermometer. Today, we have no good metrics for user distraction even though it plays a central role in pervasive computing. The same observation applies to many other attributes such as context awareness, seamlessness, translucency, privacy, and trustworthiness.

Pervasive computing is about making computing invisible. How do you demonstrate progress in invisibility? The better your system, the less there is to see!

The complexity of phenomena surrounding such attributes defies easy quantification. Yet, we can take heart from the efforts of other disciplines that deal with phenomena that are at least as complex as pervasive computing. Economics, for example, captures many facets of wealth creation and consumption though metrics such as the Gross National Product, Consumer Price Index, Dow Jones Industrial Average, and Consumer Confidence Index. In management, the Human Capital Index measures how well a company's human resources practices and policies generate shareholder value. In international relations, the UN Human Development Index measures the well-being of a nation's citizens by combining poverty, literacy, education, and life expectancy into a single index.

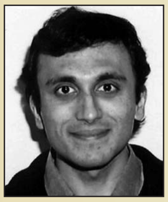
Sometimes, an operational or outcome-focused approach to defining a metric might succeed where alternative approaches fail. This is best illustrated by the following anecdote—a true story, to the best of my knowledge. In the late 1960s, Donald Knuth published the first volume of his series *The Art of Computer Programming*. In spite of paying considerable attention to detail, he knew the book might contain bugs. He therefore offered one dollar to the first finder of each bug. Indeed, readers found and reported numerous bugs, and each received a check for one dollar. Many years later, Knuth published the second edition of this work. Because finding any remaining bugs was likely to be much harder, Knuth offered two dollars to the first finder of each bug. However, between the first and second editions, Knuth had become very famous. For many people, his autograph was worth more than $2, so many saved the check as a souvenir rather than cashing it.

This suggests a metric for that elusive attribute we call fame: what is the largest amount Knuth could have offered such that some fixed fraction of the checks (say, 50 percent) would never be cashed? That dollar figure is a reasonable metric of fame: more-famous people can, presumably, write much larger checks, knowing that many of them will never be cashed.

QUANTIFYING SECURITY

How might we apply this style of reasoning to metrics for pervasive computing? Consider a property such as security. Today, no metric lets us say, "Software package A is X units more secure than B." Or, "Software package A can withstand up to X units of attack." The security community is likely to view such statements as meaningless if not misleading. Yet, intuitively, we know that

NEW EDITORIAL BOARD MEMBER



Rahul Sukthankar is a principal research scientist at Intel Research Pittsburgh and an adjunct research faculty member at the Robotics Institute, School of Computer Science at Carnegie Mellon University. His research interests include computer vision, machine learning, information retrieval, and robotics. He received his BSE in computer science and engineering from Princeton University and his PhD in robotics from Carnegie Mellon University. He's a member of the IEEE, the ACM, and AAAI. Contact him at rahuls@cs.cmu.edu; www.cs.cmu.edu/~rahuls.

a carefully crafted software package, built with great attention to detail in both algorithm design and implementation, is likely to have fewer vulnerabilities than a hastily assembled package. Why can't we quantify this distinction?

An operational approach to such quantification might proceed as follows: Use software package A to guard some secret (such as a large random number), and welcome Internet attacks on the package for some time period (say, a week). Offer a reward of $X to the first person who discovers and reports the secret. If someone reports the secret, the package is clearly not usable.

The interesting case is when no one reports the secret within the specified time period. We cannot immediately conclude that the package has no vulnerabilities. Having discovered a vulnerability, one or more successful attackers might remain silent in the hope of exploiting the vulnerability many times after the package is deployed. We have to conduct a game-theoretic analysis of an attacker's state of mind to obtain the correct inference. Suppose the reward, $X, is a large figure, and suppose package A will only be used to protect very small prizes when deployed. In that case, the attacker has more to gain by reporting success than by remaining silent—the attacker will always try to maximize his or her expected lifetime reward. We can then view the notation "Successfully protected a reward of $X on the Internet for time period T in year YYYY" as a security rating for package A. Because computing technology improves over time and attack techniques grow more sophisticated, periodic recertification will be necessary to ensure that security ratings remain meaningful.

These are, of course, highly speculative thoughts that need to be explored in depth and validate. It is clear, however, the time is ripe for defining metrics for various aspects of pervasive computing and for developing benchmarks that let us compare systems with respect to these metrics. We will then be on our way to being a true science, as characterized by Kelvin. P

REFERENCES

1. B. Noble and M. Satyanarayanan, "Trace-Based Mobile Network Emulation," *Proc. ACM Sigcomm '97 Conf.*, ACM Press, 1997, pp. 51–61.
2. L.B. Mummert, M.R. Ebling, and M. Satyanarayanan, "Exploiting Weak Connectivity for Mobile File Access," *Proc. 15th ACM Symp. Operating Systems Principles*, ACM Press, 1995, pp. 143–155.
3. S.L. Graham, P.B. Kessler, and M.K. McKusick, *gprof: a Call Graph Execution Profiler*, http://docs.freebsd.org/44doc/psd/18.gprof/paper.pdf.
4. J. Flinn and M. Satyanarayanan, "PowerScope: A Tool for Profiling the Energy Usage of Mobile Applications," *Proc. 2nd IEEE Workshop Mobile Computing Systems and Applications* (WMCSA 99), IEEE CS Press, 1999, p. 2.
5. S. Consolvo and M. Walker, "Using the Experience Sampling Method to Evaluate Ubicomp Applications," *IEEE Pervasive Computing*, vol. 2, no. 2, 2003, pp. 24–31.

IEEE Computer Society Publications Office
10662 Los Vaqueros Circle, PO Box 3014
Los Alamitos, CA 90720-1314

STAFF

Lead Editor
Shani Murray
smurray@computer.org

Group Managing Editor
Crystal R. Shif
cshif@computer.org

Senior Editors
Dale Strok, Dennis Taylor, and Linda World

Staff Editor
Rita Scanlan

Editorial Assistants
Brooke Miner and Molly Mraz

Magazine Assistant
Hilda Hosillos
pervasive@computer.org

Contributing Editors
Keri Schreiner and Joan Taylor

Design Director
Toni Van Buskirk

Layout & Technical Illustrations
Carmen Flores-Garvey and Alex Torres

Publisher
Angela Burgess
aburgess@computer.org

Assistant Publisher
Dick Price

Membership & Circulation Marketing Manager
Georgann Carter

Business Development Manager
Sandra Brown

Advertising Coordinator
Marian Anderson

Submissions: Access the IEEE Computer Society's Web-based system, Manuscript Central, at http://cs-ieee.manuscriptcentral.com/index.html. Be sure to select the right manuscript type when submitting. Articles must be original and should be approximately 5,000 words long, preferably not exceeding 10 references. Visit www.computer.org/pervasive for editorial guidelines.

Editorial: Unless otherwise stated, bylined articles as well as products and services reflect the author's or firm's opinion; inclusion does not necessarily constitute endorsement by the IEEE Computer Society or the IEEE.

IEEE pervasive COMPUTING
MOBILE AND UBIQUITOUS SYSTEMS