Helping Everyday Users Establish Confidence for Everyday Applications

Mary Shaw
Institute for Software Research, School of Computer Science
Carnegie Mellon University, Pittsburgh PA
mary.shaw@cs.cmu.edu

January 2007

Abstract

End users obtain their desired results by combining elements of information and computation from different applications. Software engineering provides little support for identifying, selecting, or combining these elements – that is, for helping end users to design computational support for their own tasks. Software engineering provides even less support to help end users to decide whether the resulting system is sufficiently dependable –whether it will meet their expectations. Many users, especially end users, base judgments about software on informal and undependable information, and they draw conclusions with informal rather than rational decision methods. We have been developing support for everyday dependability, with an emphasis on expressing expectations in abstractions familiar to the user and on obtaining software behavior that reasonably satisfies those expectations. In this Dagstuhl I would like to explore the differences between everyday informal reasoning and the rational processes of computer science in order to develop means for establishing credible indications of confidence for end users.

Everyday Dependability for Everyday Users

"Dependability" is an overarching property of software systems that includes, to various viewers and to various extents, elements of correctness, reliability, fault-tolerance, performance, security, usability (without surprises), robustness, accuracy, and numerous other properties. Everyday dependability provides enough assurance to carry out ordinary activities. Everyday software systems may be undependable, but the consequences of that undependability are not catastrophic, a human will probably notice and intervene before the effects spread, and the number of people affected is modest. For everyday software, it is generally not difficult to recover from failures.

Everyday users are not computing professionals; they create small software systems from available information and computing elements, they use abstractions drawn from their problem domains, theybare ill-equipped to evaluate the elements they use, and they are often mystified by the behavior and requirements of their software. We focus on their design and use of suites of these information and computing elements elements, applications, data sources, web pages, and other distribugted content rather than on correct programming within an application.

Within my group,

- Orna Raz explored dependability of online data feeds. The semantics of these data feeds sometimes goes awrt\y, but different users are sensitive to erroneous values to different extents. Raz developed a technique for end users to describe their individual expectations about a data feed and to translate those expectations to predicates that could monitor the data feed dynamically and adapt themselves to certain kinds of gradual systematic change in the data feeds.
- Vahe Poladian is exploring ways to provide users of mobile devices with the most satisfactory service given the limited resources of the device. Using a "task level" abstraction, he selects a

Dagstuhl Seminar Proceedings 07081
End-User Software Engineering
http://drops.dagstuhl.de/opus/volltexte/2007/1096

sequence of suites of applications that can perform the sequence of tasks planned by the user. Using models of resource availability and the resource consumption of candidate applications, he chooses the configurations that will lead to the best expected value of the user's utility.

- Chris Scaffidi is exploring abstractions that support the activities of specific identified groups of end user programmers. He refined Boehm's estimate of the number of end user programmers and collaborated with Information Week on a survey that suggested three distinct types of end users based on the kinds of abstractions they use (interestingly, they clustered by type of abstraction, not by type of application). He is developing a technique for helping users create data abstractions that correspond to the users' problems, to reformat the information as required by applications, and to express the degree of confidence a user should justifiably have that a value satisfies the expected abstraction.

Everyday Decisions

Everyday users do not have rich and robust mental models of their computing systems: they fail to do backups, misunderstand storage models (especially local and network storage), execute malware, and innocently engage in other risky behavior. The responses of computer science to the mismatch between computing systems and users' models has been to attempt to simplify the systems and to seek ways to help the users act "rationally". I hope to explore two aspects of this discrepancy at Dagstuhl.

First, we rely on "high ceremony" techniques (formal verification, systematic testing, empirical studies) for reasoning about correctness and dependability of systems. Everyday users have available a great deal of "low ceremony" evidence (reviews, reputation, informal experience, …) that is of variable accuracy and credibility. Nevertheless, many decisions are based on this sort of evidence. Can we find ways to track and manage the justifiable confidence in this evidence and to draw conclusions that are adequate for decisions about everyday software?

Second, psychologists have established that human decision-making does not adhere to rules of rational deduction, statistical analysis, and abstract reasoning about general cases. Rather, human decisions are very strongly shaped by examples, by cases that come easily to mind, and by similarity with experience. Simple linear models of a few inputs often out-perform even human experts, and there is some evidence that visual displays help with reasoning about probabilities. Can we find ways to accommodate typical human decision strategies in software design decisions, using some of the established techniques for improving informal reasoning and providing explicit indications of confidence in the results?

Acknowledgements

The work reported here is currently supported in part by the National Science Foundation (ITR-0325273) via the EUSES Consortium, by NSF Grants NSF-CCF-0438929, NSF-CNS-0613823, and NSF-CCF-0613822, and by the A.J. Perlis Chair of Computer Science at Carnegie Mellon University. The work was done in collaboration with Brad Myers, Vahe Poladian, Orna Raz, and Chris Scaffidi, and it has had the benefit of feedback from members of the EUSES Consotrium.

References

Vahe Poladian, Joao Pedro Sousa, David Garlan, and Mary Shaw. Dynamic Configuration of Resource-Aware Services. Position paper for *ICSE-2004, 26th Int"l Conf on Software Engineering*, Edinburgh, Scotland, May 2004, pp.604-613

Orna Raz, Rebecca Buchheit, Mary Shaw, Philip Koopman, and Christos Faloutsos. Automated Assistance for Eliciting User Expectations. *International Conference on Software Engineering and Knowledge Engineering (SEKE"04)*, Banff, Canada, June 2004, pp. 80-85.

Chris Scaffidi, Andrew Ko, Brad Myers, and Mary Shaw. Dimensions Characterizing Programming Feature Usage by Information Workers. *VL/HCC'06: Proceedings of the 2006 IEEE Symposium on Visual Languages and Human-Centric Computing*, pp. 59-62, 2006.

Chris Scaffidi, Mary Shaw, and Brad Myers. Games Programs Play: Obstacles to Data Reuse. *Position paper for 2nd Workshop on End User Software Engineering (WEUSE),* at the Conference on Human Factors in Computing Systems (CHI), 2006, unpaginated.

Chris Scaffidi, Mary Shaw, and Brad Myers. Estimating the Numbers of End Users and End User Programmers. *VL/HCC'05: Proceedings of the 2005 IEEE Symposium on Visual Languages and Human-Centric Computing*, 2005, pp. 207-214.

Joao Pedro Sousa, Vahe Poladian, David Garlan, Bradley Schmerl, and Mary Shaw. Task-based Adaptation for Ubiquitous Computing. in *IEEE Transactions on Systems, Man and Cybernetics - Part C: Applications and Reviews*, Vol. 36, No. 3, May 2006, pp. 328-339.