Software Performance Engineering Then and Now: A Position Paper

Connie U. Smith
Performance Engineering Services
PO Box 2640
Santa Fe, New Mexico, 87504-2640 USA
www.spe-ed.com

ABSTRACT

Software Performance Engineering (SPE) is about developing software systems that meet performance requirements. It is a proactive approach that uses quantitative techniques to predict the performance of software early in design to identify viable options and eliminate unsatisfactory ones before implementation begins. Despite its effectiveness, performance problems continue to occur. This position paper examines the evolution of SPE. It often helps to re-examine history to see if it yields insights into the future. It concludes with some thoughts about future directions.

General Terms

Software Performance Engineering, Performance Modeling, Performance Requirements

1. INTRODUCTION

Software Performance Engineering (SPE) [3] is about developing software systems that meet performance requirements. It is a proactive approach that uses quantitative techniques to predict the performance of software early in design to identify viable options and eliminate unsatisfactory ones before implementation begins. Software performance models enable the evaluation of trade-offs in software functions, hardware size, quality of results, and resource requirements. This is the heart of the methodology; it also incorporates principles and patterns for designing responsive systems, methods for conducting evaluations, testing for verification and validation that performance requirements will be met, strategies for dealing with uncertainty, etc. SPE is neither a new nor a revolutionary approach. It applies proven techniques to predict the performance of emerging software and systems and respond to problems while they can be fixed with a minimum of time and expense.

Despite its effectiveness, performance problems continue to occur. The most notable in recent times was the role out of the healthcare.gov web site and the dramatic performance failures that resulted. They expected thousands of people to sign up on the first day, but the performance limitations meant fewer than 10 actually succeeded. That is just one of many documented disasters, but it rose to the level of US Congressional inquiry.

This position paper examines the evolution of SPE. It often helps to re-examine history to see if it yields insights into the future. Therefore we examine the origin of SPE, its development and characteristics over time, its current state of the art and state of the practice, and what has changed. It concludes with some thoughts about future directions.

WOSP-C'15, January 31 2015, Austin, TX, USA.

ACM 978-1-4503-3340-5/15/01 ...$15.00.
http://dx.doi.org/10.1145/2693561.2693567.

2. SPE ORIGINS

Addressing performance throughout software development is not a revolutionary proposition; developers routinely sought performance in the early years of computing [2]. The space and time required by programs had to be carefully managed so that those programs would fit on very small machines. The hardware grew, but rather than eliminating performance problems, it made larger, more complex software feasible, and programs became systems of programs. Performance modeling and assessment of these early systems was labor-intensive and thus expensive. Analysts used handcrafted simulation models. Consequently creating and solving models was time-consuming, and keeping models up to date with the current state of evolving software systems was also problematic. Thus, modeling and evaluation were cost-effective for only those systems, such as flight-control systems and other mission-critical embedded systems that had strict performance requirements. Systems without critical performance requirements adopted the "fix-it-later" method. The premises of fix-it-later were:

- Performance problems are rare
- Hardware is fast and inexpensive
- It is too expensive to build responsive systems
- You can tune software later, if necessary
- Efficiency implies "tricky code" which causes maintenance problems

In the late 1970s all of these premises became false. Resulting performance failures were dramatic with the rapid increases in on-line users and thus the visibility of performance problems. The new reality of the fix-it-later premise became:

> Even though hardware was relatively fast and inexpensive, it was not free. When many new systems required

additional, unpredicted hardware expenditures, businesses were dissatisfied.

SPE, the availability of modeling tools, techniques, and experts actually increased productivity over fix-it-later by preventing problems that delay delivery and by preventing tricky-code maintenance problems.

Tuning can always improve performance, but not as much as appropriate design can, and tuning changes may require significant implementation efforts that often delay the delivery of urgently-needed software. Tuning improvements are usually modest compared to the dramatic improvements achievable with re-design.

In the early days of computing, hand-crafted code was required for efficiency. Responsiveness is not the same as efficient use of computer resources. So responsiveness could be achieved by using resources (e.g., caching information likely to be requested) for responsiveness (e.g., when it is requested). It was not necessary to introduce "tricky code" to achieve efficiency.

The SPE approach gained momentum and leading edge companies developing new, large-scale systems adopted SPE. This was especially true of companies implementing "bleeding-edge" technology that had not been tried before. There was no handbook for how to prevent performance problems, and the resulting performance was visible to a larger and larger number of users.

3. SPE DEVELOPMENT

From my perspective as an SPE researcher, educator, and consultant, there was a surge of interest and application of SPE in the 1980s and 90s and into the early 2000s. Classes were full, and many companies adopted the proactive SPE techniques. When progressive organizations began a new large-scale project with new technology with which they had no experience, SPE was one of the first technologies they explored. Occasionally organizations that were not early adopters created new systems, experienced performance problems with their first release, and brought in SPE experts to help them identify and fix problems.

4. NOW

Many organizations continue to use SPE as they always have. Others have experienced performance problems and have adopted SPE to correct them. There is still a strong research component investigating improved SPE modeling and other techniques. In general, though, there are fewer organizations using SPE for new development.

Despite the availability of new development methods especially Model Driven Engineering (MDE), SPE, and other improvements, a majority of government and other systems fail to meet requirements on initial delivery. Consider the data in Figure 1. It is a compilation of data from two separate investigations one in 1979 and one in 1995. I have not found more recent published study data, however I have informally shown this data to many involved in government system contracting and development and they believe there is not much difference today.

The number of experts with the ability to manage performance during development is dwindling (as evidenced by

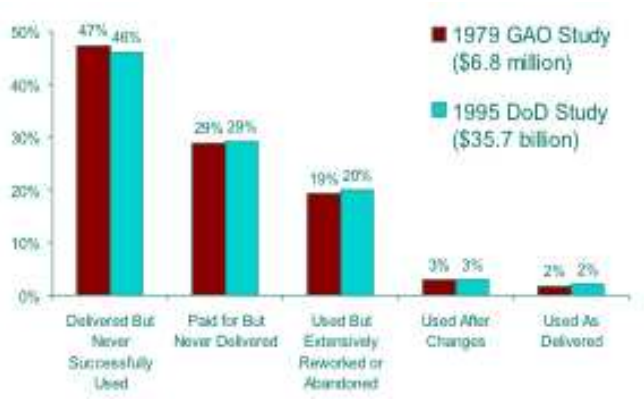

Figure 1: Success Data for Government Projects

attendance at performance specialist conferences and membership in related organizations). Figure 2 shows the attendance over time at Computer Measurement Group (CMG) conferences. Note that this data is approximate and experiential but the trends are accurate. Performance testers are still in demand, but more often they are called in after performance failures rather than brought in during development when changes are easier and problems can be prevented.

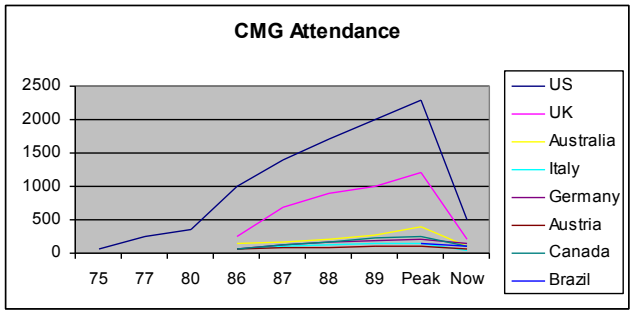

Figure 2: CMG Attendance

Today, organizations still have significant performance problems, but either most are not newsworthy or users cope with poor performance as if it is expected. Perhaps there are fewer totally new systems with new technology being created in IT. There is less interest in the proactive approach to managing performance. Most performance efforts these days are focused on measurement and testing.

There are also fewer individuals with expertise in SPE. The previous generation is retiring and they are not being replaced by new people with the same skills. In fact, there are fewer universities in the US that have courses with depth of coverage in performance modeling and other SPE skills.

5. FUTURE DIRECTIONS

What should the path forward look like? Are we now at a phase where the premises for fix-it-later are valid again?

There is still a need for proactive SPE on new system development. Embedded and control systems have become pervasive and they are not like the old ones that had narrow special purposes and were self contained. Now the control systems have touch screen user interfaces, talk to the inter-

net, and control devices that for years have had only mechanical controls. The software is far more complex, and it is being implemented on traditional operating systems such as Windows Embedded.
Some challenges include:

- Better educational coverage is needed. In addition to performance fundamentals, performance modeling and tools in engineering and computer science, business schools need coverage of performance management for projects, case studies, business cases, etc.
- New practitioners seem to be "reinventing the wheel", even basic performance measurement and testing techniques. Some of the fundamental performance texts are out of print.
- Many performance tools are no longer available.
- Many new software development efforts are outsourced; if contractors are not required to use SPE methods, they are not likely to do so.

Ken Kolence, the first recipient of CMGs A.A. Michelson lifetime achievement award for his work in software physics, long ago told me it would be necessary to have a "shrink wrap" solution for SPE adoption. He said it is too difficult to sell a philosophy. Robert Goldberg, co-founder of BGS the company behind one of the first successful performance modeling tools (BEST/1) advised that selling SPE adoption requires finding someone in the organization who is motivated to "buy insurance that their project will not fail," such as a financial officer, user organizations, etc.
It is clear that we need more and better tools to make it easier to apply SPE techniques and require less performance expertise than in the past. This has been a focus of my recent research. Organizations also need to know that proactive performance management is possible and the preferable way to develop new systems.

One last topic for consideration: Insularity revisited. In 1986, Domenico Ferrari [1] proposed that performance evaluation was too isolated. We had separate courses in performance, separate conferences and publications that were not integrated with the other areas of computer science that needed it. He proposed that it would be better to incorporate performance topics in other courses concerned with design and implementation, conferences and publications. Today some of the more successful university programs in performance are insular. It seems that a critical mass of effort is needed to make significant advances. Those with only a few members who attempt to integrate their work with others seem less successful. Integration into other courses is problematic when professors do not have sufficient background in performance engineering so this becomes a "chicken and egg" problem. Or was Ferrari right, and the insularity is what caused the decline in performance evaluation in many universities and in practice?

6. REFERENCES

[1] D. Ferrari. Considerations on the insularity of performance evaluation. *IEEE Transactions on Software Engineering*, 14:21–32, June 1986.

[2] C. Smith. *Performance Engineering of Software Systems*. Addison-Wesley, 1990.

[3] C. Smith and L. Williams. *Performance Solutions: A Practical Guide to Creating Responsive, Scalable Software*. Addison-Wesley, 2002.