

Introduction to “Engineering of Software: The Continuing Contributions of Leon J. Osterweil”

Peri L. Tarr and Alexander L. Wolf

Peri L. Tarr is at IBM Thomas J. Watson Research Center, P. O. Box 704, Yorktown Heights, NY 10598 USA, tarr@us.ibm.com

Alexander L. Wolf is at Imperial College London, Department of Computing, 180 Queen's Gate, London SW7 2AZ United Kingdom, a.wolf@imperial.ac.uk

Abstract Software engineering research can trace its roots to a small number of highly influential individuals. Among that select group is Prof. Leon J. Osterweil, whose work has fundamentally defined or impacted major directions in software analysis, development tools and environments, and software process. His exceptional and sustained contributions to the field have been recognized with numerous awards and honors throughout his career. This section briefly reviews his exceptionally distinguished career.

Software pervades our lives today, powering everything from PlayStations® to the Space Station. Entire governments and economies run on software. Businesses differentiate themselves by the quality of their software services. Students learn using software. Food and automobiles and pharmaceuticals are produced using software. Medical devices that give people with a myriad of conditions a new lease on life are powered by software. People collaborate, stay connected, and reap “the wisdom of crowds” [7] using software. It is hard to imagine life without software.

But it wasn't always so. The field of software engineering started from very humble origins as relatively simple aids to productivity, such as mathematical processing packages and small-scale applications to help with back-office functions. Very few could ever have conceived that, one day, the engineering of software would become one of the most important, impactful activities in the world.

Leon J. Osterweil is one of those few. Lee saw past software as a limited aid to corporate and personal productivity, and into a future where it would be possible to engineer large-scale, reliable software that could enhance peoples' effectiveness and enrich their lives. And he went further: he hypothesized that the software processes by which software is engineered could themselves be engineered to help build those game-changing software systems. This doesn't sound like a vision today—it is the reality in which we live—but in the 1970s, when Lee started his exceptionally distinguished career, it was prescient.

Lee has been a major force in driving software engineering from vision to reality. For more than three decades, he has fundamentally defined or significantly af-

fectured major approaches to software analysis, development tools and environments, and software process—all critical parts of the software engineering discipline as it is practiced today. The depth and breadth of his work is seen in research, government, academia, and industry. Much of his work and many of his ideas continue to have major influence today—a truly remarkable statement about work in a field in which advances are old in a matter of months or a small number of years. Lee has driven and influenced the field of software engineering in both obvious and subtle ways. He has defined and developed many of the core ideas in the field, and he has also nurtured generations of its practitioners.

Lee started his career in mathematics, receiving the A.B. degree from Princeton University in 1965, and the M.A. and Ph.D. degrees from the University of Maryland in 1970 and 1971, respectively. After graduate school, he became a professor at the University of Colorado at Boulder, where he ultimately served as chair of the Department of Computer Science. During his tenure in Boulder, he and his students produced multiple seminal pieces of research in the analysis of software. Part I of this volume examines the significant and long-lived impact of that work.

It was also during his time in Boulder that Lee’s vision of software engineering as a discipline began. After laying the foundation for a study of what would eventually be called “software development environments” with his work on Toolpack (Chapter 11) and Odin (Chapter 12), Lee co-founded the Arcadia Consortium in 1987. Arcadia reflected the group’s vision of the engineering of software on a grand scale. Its goal was to perform the research and proof-of-concept development required to make fully integrated, end-to-end software development environments—ones that support all aspects of the engineering of software—a reality. Arcadia was a decade-long joint effort of researchers at the University of Colorado at Boulder, the University of California at Irvine, and the University of Massachusetts at Amherst, and it included participants from TRW, Incremental Systems Corporation, and others. The sheer number and extent of the contributions that came out of Arcadia is overwhelming. Part II examines some of Lee’s major contributions in this area.

It was during this time when Lee had the insight that “software processes are software too,” [3] which led to his most profound contributions to software engineering to date. In particular, he noticed that the different processes carried out during the course of the engineering of software suffered from many of the same characteristics and problems that software itself does. Processes are frequently erroneously specified, unclear in intent, difficult to debug, and even more difficult to understand. This led him to hypothesize that programming software processes, and subjecting their development to the same, rigorous methods, analyses, and testing to which software is subjected, could benefit processes the same way it had benefited software. Lee’s vision radically altered the whole software environments landscape. Even today, *process-centered* software environments—a hallmark of Arcadia—are just emerging in the state-of-the-practice. Part III looks Lee’s profound and sustained contributions in the area of software processes.

In 1988, Lee joined the faculty of the University of California at Irvine, where he also served as chair of the Department of Information and Computer Science. While at Irvine, he co-founded the Irvine Research Unit in Software (IRUS), which seeks to advance the state-of-the-art and state-of-the-practice in software engineering by promoting partnerships between academic and industrial participants. IRUS—which continues its invaluable work to this day—is characteristic of Lee: he builds bridges between people with different knowledge and expertise to enable them to find solutions together that none would have found individually.

In 1993, Lee joined the faculty at University of Massachusetts at Amherst, where he remains today. He served as Interim Dean of the College of Natural Sciences and Mathematics from 2001 to 2005. He serves as co-director for the Laboratory for Advanced Software Engineering Research (LASER) and for the Electronic Enterprise Institute. In recent years, he has expanded his research on process definition and analysis to some important domains, including medical safety, dispute resolution, electronic commerce, and e-government, and he has continued to make significant contributions in software testing and analysis.

The pivotal role that software plays in our society means that software engineering practice desperately needs a strong research base on which to draw. The questions inevitably arose: how well have its results supported software engineering practice, and how can it best do so in the future? In 2002, Lee decided to take on these critical questions. He enlisted the help of three colleagues to form the Impact Project [5]. His vision was to produce a series of rigorous, scholarly, and objective studies in various subfields of software engineering, with the goals of assessing past impact and making recommendations to assure future impact. To date, three such studies have been published as peer-reviewed journal articles, authored by teams from both industry and academia, and more are underway. These studies cover the areas of software configuration management [2], modern programming languages [6], and middleware technology [1]. They clearly identify how synergisms among different areas of computer science (e.g., programming languages, distributed systems, database management) and healthy exchanges between academia and industry have sparked key innovation. The Impact Project has been remarkably successful and, indeed, impactful, in affecting both the perception of software engineering research and research agendas in the field.

The Impact Project is just one example of the key and sustained leadership role in the software engineering community that Lee has played throughout his career. His outstanding research contributions have been recognized and honored with awards too numerous to list exhaustively. Lee is a Fellow of the ACM, and he has received the ACM SIGSOFT Outstanding Research Award in recognition of his extensive and sustained research impact. His seminal paper on software process programming was awarded the Most Influential Paper of ICSE 1987, which recognizes it as the paper from that conference which had the most significant impact on the field over the decade that followed its publication. His ICSE 1986 paper on tool integration was selected as the runner-up for the same honor. Lee has served as General Chair for FSE 1998 and ICSE 2006, and Program Chair for

many conferences in different areas of software engineering, including ICSE 1994. Lee has been an ACM Lecturer, has been on the board of ACM's Transactions on Software Engineering and Methodology, and has been a member of the Software Engineering Institute's Process Program Advisory Board. He has served on technical advisory boards for, or consulted with, several industrial organizations, including AT&T, Boeing, IBM, SAIC, and TRW.

Lee's research contributions have been so extensive that some people are surprised to learn that he is also a very talented and dedicated educator. He has directed the research of many Ph.D. students who have themselves gone on to achieve significant impact and receive major awards. Lee has also mentored generations of young researchers, guiding them in their careers. He was recently honored with the ACM SIGSOFT Most Influential Educator Award, in recognition of his career-long achievements as an educator and mentor.

Lee's expertise has earned him prominent roles on National Academies Panels, which provide advice to the United States government on key topics in science, engineering, and medicine. In 2007, he co-authored an assessment of the Social Security Administration's electronic service provision [3]. He is currently a member of a National Academies Panel on "Future Information Architectures, Processes, and Strategies for the Centers for Medicare and Medicaid Services."

It has been a privilege and a delight to have the opportunity to assemble the papers in this volume, and to reflect on the distinguished career of someone who laid so much of the foundation for the field of software engineering, and who has done—and continues doing—so much in his career to promote the vitality of the field, its practitioners, and all those whose lives are enriched by it. We hope you enjoy and benefit from this volume as much as we have.

Acknowledgments We are very grateful to Lori Clarke, Stan Sutton, and Clay Williams for their outstanding feedback and input.

References

- [1] Emmerich W, Aoyama M, Sventek, J (2008) The Impact of Research on the Development of Middleware Technology. *ACM Trans. Softw. Eng. Methodol.* 17(4)
- [2] Estublier J, Leblang D, van der Hoek A, Conradi R, Clemm G, Tichy W, Wiborg-Weber D (2005) Impact of software engineering research on the practice of software configuration management. *ACM Trans. Softw. Eng. Methodol.* 14(4)
- [3] Osterweil L (1987) Software Processes are Software Too. In *Proc. International Conference on Software Engineering*
- [4] Osterweil L, Millet L, Winston J (2007) Social Security Administration Electronic Service Provision: A Strategic Assessment. National Academies Press
- [5] Osterweil LJ, Ghezzi C, Kramer J, Wolf AL (2008) Determining the Impact of Software Engineering Research on Practice. *IEEE Computer* 41(3)
- [6] Ryder BG, Soffa ML, Burnett, M (2005) The impact of software engineering research on modern programming languages. *ACM Trans. Softw. Eng. Methodol.* 14(4)
- [7] Surowiecki, J (2005) *The Wisdom of Crowds*. Anchor

Engineering of Software

The Continuing Contributions of Leon J. Osterweil

Tarr, P.L.; Wolf, A.L. (Eds.)

2011, VIII, 417 p., Hardcover

ISBN: 978-3-642-19822-9