Foreword

This book, *Agile Estimation Techniques and Innovative Approaches to Software Process Improvement*, appears at an opportune time, when the area of Software Process Improvement is going through transformational stages with significant innovative approaches. In this Foreword, we extract from a presentation made at Zacatech 1.0 and expand on it to incorporate new innovative initiatives in software process improvement presented at the SEPG Europe 2012. The text chronicles the evolution of the area prior to and after the creation of the Software Engineering Institute with the introduction and adoption of capability maturity models worldwide. It then refers to agile methodologies, which are now seeing increasing adoption in the improvement of the software development process. It culminates with recent initiatives at the SEI, which deal with new approaches to software engineering process improvement.

Software Process Improvement has been a recurrent theme in Software Engineering for many years, going back to the time when the need for the management of the software development process was identified as a major problem area, circa 1982, and later, in 1983, in the DoD Software Technology for Adaptable, Reliable Systems (STARS) strategy. Earlier, there were very few software engineering practices that produced consistent, well-documented, or well-understood results. Supporting software tools were often ad-hoc. Yet, successful software development programs and a growing body of literature were beginning to emerge. IBM had mature efforts that had proven effective. When, in 1984, the Software Engineering Institute (SEI) was founded, the theme was reemphasized. The SEI strategic plan—supported by DoD and defense contractors—was recognized as a fundamental activity. The SEI recruited Watts Humphrey from IBM and the Process Management Framework Project was created in 1986. In order to elicit a consensus on practices leading to improved software development, under the auspices of the Air Force Program Manager, the SEI conducted a study of “best practices,” and workshops were held with leading software professionals in the defense industry, commercial industry, and academia. It was at this time that a number of Software Process Improvement Networks, SPINs, were created in the software engineering community worldwide. The outcome of the workshops was the enumeration of 18 practices to use as models for the work of organizations, and a subsequent SEI Maturity Questionnaire identified 18 key process areas and a 5-level model of organizational maturity based on the implementation of the process areas. More precise definitions of the practices and the model resulted in seminal contributions of the SEI, to wit: *Software Capability Maturity Model in 1991; Capability Maturity Model for Software, Version 1.1*. The following technical reports and/or books attracted considerable attention in the software community: *Managing the Software Process; Software Engineering Process Group Guide; An Analysis of SEI Software Process Assessment Results: 1987-1991*.

In response to a request of the Air Force for establishing ways for the Acquisition Offices to assess the maturity of their contractors, the SEI then developed the Software Capability Evaluation (SCE).
Training and documentation supported its use by the DOD acquisition community. The SCE method was widely used in software-intensive systems acquisitions by the organizations serving the DOD. This provided an incentive for the use of the SEI’s CMM to achieve improvements in management and technical practices. The evolution of this work is a de-facto standard for evaluating and improving process management in software and systems engineering. After the first CMM for use in software development, the SEI proceeded to create a powerful framework for processes in multiple disciplines, which eventually led to the development of maturity models for other areas: People CMM, for managing human assets; Systems Engineering CMM; and eventually in the cyber security area, the CERT Resilience Management Model, released in 2010.

A landmark in the evolution of maturity models was the Capability Maturity Model Integration (CMMI), developed to improve the usability of maturity models by integrating many different models into one framework. Currently, CMMI models are available in three areas: product and service development (CMMI for Development); service establishment, management, and delivery (CMMI for Services); and product and service acquisition (CMMI for Acquisition). The software engineering, management and measurement processes defined by CMMI have seen widespread adoption with implementations in 74 countries on 6 continents. The CMMI has reached a level of maturity itself, which precludes further research supported by the DOD. Consequently, while recognizing its importance and future developments, it has now been transferred from the SEI to the CMMI Institute, a separate entity of Carnegie Mellon University.

Recognizing the great impact that the methodology and principles of the Capability Maturity Model had in the software engineering process improvement, they were extended to the people who actually do the work—the practicing engineers. The Personal Software Process (PSP) was built on the principle that every engineer who works on a software system must do quality work to produce a quality system. The development of the PSP began with the application of CMM principles to writing small software programs. Further refinement led to a disciplined personal framework. This allows engineers to establish and commit to effective engineering and management practices for their software projects.

Recognizing that effective engineers must be able to work on teams, the Team Software Process (TSP) was developed to help development teams establish a mature and disciplined engineering practice toward producing secure, reliable software in less time and at lower costs. The TSP has been applied in small and large organizations in a variety of domains with well-documented results: productivity improvements of 25% or more, testing costs and schedule reductions of up to 80%, and cost savings of 25-50% per software product release.

Agile software development (http://en.wikipedia.org/wiki/Agile_software_development) refers to a group of software development methods based on iterative and incremental development, where requirements and solutions evolve through the collaboration between self-organizing, cross-functional teams. It promotes adaptive planning, evolutionary development and delivery, and a time-boxed iterative approach, and encourages rapid and flexible response to change. It is a conceptual framework that promotes foreseen interactions throughout the development cycle. The Agile Manifesto introduced the term in 2001 (http://agilemanifesto.org/).

So-called lightweight agile software development methods evolved in the mid-1990s as a reaction against heavyweight waterfall methods, which were characterized by their critics as a heavily regulated, regimented, micromanaged, waterfall model of software development. Proponents of lightweight agile methods contend that they are a return to development practices from early in the history of software development.

Looking to the future, the SEI has established the following initiatives (SEI Blog, 2012): Innovating Software for Competitive Advantage, Securing the Cyber Infrastructure, Accelerating Assured Software Delivery for the Mission, Advancing Quantitative Methods for Engineering Software. These initiatives reflect the increasing role that agile methods are playing in the research and development funded by the SEI.

In Innovating Software for Competitive Advantage, the area focuses on creating innovations to revolutionize the development of secure systems-of-systems. Examples of projects are incremental and iterative architecture for agile software, architecture patterns to secure key quality attributes in systems-of-systems, and optimizing resources for mobile platforms at the edge.

In Securing the Cyber Infrastructure, the area focuses on enabling with well-informed confidence the use of ICT technologies to guaranty a secure connected world. Examples of projects are static analysis checker for C/C++ coding patterns and patterns for architecting enterprise IT systems to improve resilience against insider threats.

In Accelerating Assured Software Delivery for the Mission, the area focuses on guarantying a predictable return in the mission of the acquisition, operation, and maintenance of resilient software. A challenge here would be to produce for the DOD an empirical base to establish opportunities, conditions, and thresholds for the use of agile methods. Examples of Projects are patterns for joint acquisition programs and contingency models to evaluate the applicability of the agility in resilient software systems in critical missions.

In Advancing Quantitative Methods for Engineering Software, the area focuses on the sustainability, affordability, and availability of software-reliant systems through data-driven models, measurement, and management methods. This area has some challenges, like improving costs at the beginning of the life cycle in the acquisition and being able to construct incrementally agile methods in software development. Examples of Projects are methods and tools for estimating probabilistically software costs and models and methods of engineering, management, and measurements practices.

Angel G. Jordan
Carnegie Mellon University, USA
presentations, and reports. Dr. Jordan is active in national and international science and technology research and education issues, does extensive consulting work, serves on various boards in the U.S. and abroad, and holds memberships in prestigious engineering and science associations, such as the National Academy of Engineering in the US and the Real Academia de Ingeniería in Spain. He is a Fellow of the IEEE and the AAAS. He was the founder of the Software Engineering Institute, where he has been twice Acting Director, and co-founder of the Robotics Institute. He is Doctor Honoris Causa of the Universidad Politécnica de Madrid, the Universidad Pública de Navarra, and the Universidad Carlos III.

REFERENCES