Advances and Challenges of Software Engineering

Stephen S. Yau
Information Assurance Center, Department of Computer Science and Engineering
School of Computing and Informatics, Ira A. Fulton School of Engineering
Arizona State University, Tempe, AZ 85287-8809, USA
yau@asu.edu

Since the invention of modern computers, the continuous advances of computer hardware and software technology have made computing systems inseparable from various critical infrastructures, social and economical activities, and our daily lives. The large variety of computer applications heavily depends on the quality and performance of software, and the cost of software development and maintenance has greatly exceeded that of hardware.

However, software engineering did not attract much attention until many difficulties have occurred in the construction of increasingly large-scale and complex computing systems in 1960’s. The so-called software crisis, which represents a set of problems on controlling the cost and schedule of software development, ensuring the quality of software, and maintaining complex software, has led to the first NATO Software Engineering Conference in 1968 to seek solutions to these problems. Over the forty years after this conference, the following are among the major advances in software engineering [1, 2]:

- Evolvement of software processes from rigid, sequential and slow processes (e.g. Waterfall processes) to iterative and agile processes (e.g. extreme programming), and standardization of process models, such as CMMI (Capability Maturity Model Integration) and ISO 12207.
- Programming paradigms and software design techniques, such as object-oriented programming and model-driven architecture.
- Various techniques for software reuse, such as design patterns and component-based software development.
- Formal methods for software specification and verification, and numerous methods for software testing.
- Software architecture, such as event-driven architecture and service-oriented architecture, and architecture description languages.

All these advances aim at improving the quality of software, reducing cost, and promoting software reuse. Despite the advances achieved so far, software engineering is still a rapidly evolving engineering discipline due to new applications derived from various new technologies, such as internet, mobile networks, sensors, and ubiquitous, grid and service computing. Many software systems are now expected to operate in increasingly heterogeneous, dynamic, and geographically dispersed environments comprising of computing devices that may not be fully trusted. Hence, there is increasing demand on security, dependability, situation awareness, adaptability, and composability of software [3]. Software needs to be secure and dependable to protect its users from malicious activities. Software needs to be situation-aware and adaptive so that it is aware of its own behavior and changes in its environment, and capable of changing its own behavior accordingly. In addition, a software system should be rapidly built by composing available components, such as Web Services, and can be used to compose more complex systems.

To address these challenges, fundamental breakthrough in software engineering research is needed, such as new knowledge on the dynamics of software-intensive systems to facilitate development of QoS-aware software, new models and techniques for automated composition of software, and new quantitative metrics for software quality.

References