SOFTWARE ENGINEERING APPLICATIONS IN SCIENCE AND INDUSTRY

DOOSTANI ALI

BSc of Computer Engineering-Software, Islamic Azad University, Masjed Soleyman
E-mail: doostaniali6@gmail.com

Abstract - Since the dawn of electronic computing in the 1940s, computing systems and their applications have evolved at a staggering rate. Software plays a central and underpinning role in almost all aspects of daily life: communications, government, manufacturing, banking and finance, education, transportation, entertainment, medicine, agriculture, and law. The number, size, and application domains of computer programs have grown dramatically; as a result, huge sums are being spent on software development . Most people's lives and livelihoods depend on this development's effectiveness. Software products help us to be more efficient and productive. They provide information, make us more effective problem solvers, and provide us with safer, more flexible, and less confining work, entertainment, and recreation environments.

Index Terms - Computer Engineering, Software

I. INTRODUCTION

The term "software engineering" has now become widely used in industry, government, and academia. Hundreds of thousands of computing professionals go by the title "software engineer"; numerous publications, groups and organizations, and professional conferences use the term software engineering in their names; and many educational courses and programs on software engineering are available. Unfortunately,as with the term "engineer" itself, the term software engineer is not always used to mean"a software engineering professional," although that is the meaning that is assumedthroughout this document. The first computer engineering degree program in the United States was established at Case Western Reserve University in 1972. As of 2015, there were 238 ABET-accredited computer engineering programs in the US.[6] In Europe, accreditation of computer engineering schools is done by a variety of agencies part of the EQANIE network. Due to increasing job requirements for engineers who can concurrently design hardware, software, firmware, and manage all forms of computer systems used in industry, some tertiary institutions around the world offer a bachelor's degree generally called computer engineering. Both computer engineering and electronic engineering programs include analog and digital circuit design in their curriculum. As with most engineering disciplines, having a sound knowledge of mathematics and science is necessary for computer engineers. Over this same period, although the software engineering discipline has evolved, thecontext has changed as well. In the 1960s, a software product was usually created as asingle, monolithic entity, executed on a computer with the support of a fairly basicoperating system. Such a product had external operations that were mainly confined tobasic file input/output. In contrast, a software system developed in today may well reusemajor components of other systems, execute on multiple machines and platforms, andinteract with other, globally distributed systems Software engineering concepts in the early years were largely dominated by the idea of structure, both in terms of the product and the processes that were recommended as themeans of creating that product.

II. COMPUTER ENGINEERING

Computer engineering is a discipline that integrates several fields of electrical engineering and computer science required to developcomputer hardware and software.[1] Computer engineers usually have training in electronic engineering (or electrical engineering),software design, and hardware-software integration instead of only software engineering or electronic engineering. Computer engineers are involved in many hardware and software aspects of computing, from the design of individual microcontrollers, microprocessors,personal computers, and supercomputers, to circuit design.

This field of engineering not only focuses on how computer systems themselves work, but also how they integrate into the larger picture.[2] Usual tasks involving computer engineers include writing softwareand firmware for embedded microcontrollers , designing VLSI chips, designing analog sensors, designing mixed signal circuit boards, and designing operating systems. Computer engineers are also suited for robotics research, which relies heavily on using digital systems to control and monitor electrical systems like motors, communications, and sensors. In many institutions, computer engineering students are allowed to choose areas of in-depth study in their junior and senior year, because the full breadth of knowledge used in the design and application of computers is beyond the scope of an undergraduate degree. Other institutions may require engineering students to complete one year of General Engineering before declaring computer engineering as their primary focus.

III. SOFTWARE ENGINEERING AS A COMPUTING DISCIPLINE

In the early days of computing, computer scientists produced software, and computerengineers built the hardware to host its execution. As the size, complexity, role, andcritical importance of software artifacts grew so did the need to ensure that they performed as intended. By the early 1970s, it was apparent that proper software development practices required more than just the underlying principles of computerscience; they needed both the analytical and descriptive tools developed within computer science and the rigor that the engineering disciplines bring to the reliability and trustworthiness of the artifacts they engineer. Drawing on computing as one of its foundations, software engineering seeks to develop and use systematic models and reliable techniques to produce high-quality software. These concerns extend from theory and principles to the development practices that are most visible to those outside the discipline. Although it is unlikely that every software engineer will have deep expertise in all aspects of computing, a general understanding of each aspect's relevance and some expertise in particular aspects are a necessity. Typical formal definitions of software engineering are:

"research, design, develop, and test operating systems-level software, compilers, and network distribution software for medical, industrial, military, communications, aerospace, business, scientific, and general computing applications."

"the systematic application of scientific and technological knowledge, methods, and experience to the design, implementation, testing, and documentation of software";

"the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance ofsoftware";

"an engineering discipline that is concerned with all aspects of software production"; and "the establishment and use of sound engineering principles in order to economically obtain software that is reliable and works efficiently on real machines.

IV. SOFTWARE ENGINEERING AS AN ENGINEERING DISCIPLINE

The study and practice of software engineering is influenced both by its roots in computer science and its emergence as an engineering discipline. A significant amount of current software engineering research is conducted within the context of computer science and computing departments or colleges. Similarly, software engineering degree programs arebeing developed by such academic units as well as in engineering colleges. Thus,software engineering maintains a stronger connection to its underlying discipline(computer science) than may be the case for some other engineering fields. In the process of constructing this volume, particular attention has been paid to incorporatingengineering practices into software development so as to distinguish softwareengineeringcurricula from those appropriate to computer science degree programs. To prepare for themore detailed development of these ideas, this section considers how the engineering methodology applies to software development. Some critical characteristics common to every other engineering discipline are equally applicable to software engineering. Thus, they have influenced both the development ofsoftware engineering and the contents of this volume.

1. Whereas scientists observe and study existing behaviors and then develop models to describe them, engineers use such models as a starting point for designing and developing technologies that enable new forms of behavior.
2. Engineers proceed by making a series of decisions, carefully evaluating options, andchoosing an approach at each decision point that is appropriate for the current task inthe current context. Appropriateness can be judged by trade-off analysis, whichbalances costs against benefits.
3. Engineers measure things, and when appropriate, work quantitatively. They calibrateand validate their measurements, and they use approximations based on experienceand empirical data.
4. Engineers emphasize the use of a disciplined process when creating and implementing designs and can operate effectively as part of a team in doing so.
5. Engineers can have multiple roles: research, development, design, production, testing, construction, operations, and management in addition to others such as sales,consulting, and teaching.
6. Engineers use tools to apply processes systematically. Therefore, the choice and useof appropriate tools is a key aspect of engineering.
7. Engineers, via their professional societies, advance by the development and validation of principles, standards, and best practices.
8. Engineers reuse designs and design artifacts.

Although strong similarities exist between software engineering and more traditional engineering, there are also some differences (not necessarily to the detriment of softwareengineering):

- Software engineering's foundations are primarily in computer science, not natural sciences.
- Software engineering models make more use of discrete than continuous mathematics.
- The concentration is on abstract/logical entities instead of concrete/physical artifacts.

- There is no manufacturing phase in the traditional sense.
- Software maintenance primarily refers to continued development, or evolution, andnot to conventional wear and tear.
- Software engineering is not always viewed as a "professional" activity. One concern for these curriculum guidelines is to help with the evolution of software engineering toward a more "professional" status.In using the term engineer and engineering extensively, this document is about thedesign, development, and implementation of undergraduate software engineering curricula.

CONCLUSION

Software companies and corporate have to take a lead in helping academia to shape the curriculum so that software engineers who can tackle the industry challenges are prepared. Academia on the other hand should analyze and incorporate changes constantly so that technology changes can be reflected as required. Both universities and institutions on one hand and companies on the other hand have to work in tandem to ensure the grooming of students who will be software engineers of tomorrow. Industry would be the first to gain from helping shape future employees as they need to endure less training costs and would get well equipped software engineers. Problems and Programmers represents a first attempt at using a physical card game toteach students about the software engineering process. It addresses many of the weaknesses ofmore traditional techniques and brings additional benefits in the form of face-to-face learningand enjoyable play. When used in conjunction with lectures and projects, Problems and Programmersallows students to gain a thorough understanding of real-world lessons that mightother wise have been poorly understood or overlooked altogether. In the same way, students and instructors need frequent feedback about the learning experience in courses. When the feedback is negative, either the student or the instructor (or both) need to take immediate action to keep the course on track. Test-first teaching is an effective method to use at 2 levels: creation and evaluation.

REFERENCES

[1] Beecham S. et al., "Motivation in Software Engineering: ASystematic Literature Review," Information & Software Technology, vol. 50, 2008,pp. 860–878.

[2] Bloom, B.S. , ed., Taxonomy of Educational Objectives: The Classificationof Educational Goals: Handbook I, Cognitive Domain, Longmans, 1956.

[3] B.W. Boehm, "A Spiral Model of Software Development andEnhancement," Computer, vol. 21, no. 5, 1988, pp. 61–72.

[4] Boehm B.W. and R. Turner, Balancing Agility and Discipline: AGuide for the Perplexed, Pearson Addison-Wesley, 2003.

[5] Brooks, F.P. and Jr. "No Silver Bullet: Essence and Accidents of SoftwareEngineering," Computer, vol. 20, no. 4, 1987, pp. 10–19.

[6] Rogers, Y. and H. Sharp H., and J. Preece, Interaction Design: BeyondHuman-Computer Interaction, 3rd ed., Wiley, 2011.

[7] Schwaber, K. and Agile Project Management with Scrum, Microsoft Press,2004.

[8] http://www.iitb.ac.in/ as on 12/05/09

[9] http://www.iitm.ac.in/ as on 12/05/09

[10] http://www.aicte.ernet.in/ as on 12/05/09

★★★