Exploration and Practice of Software Engineering Core Curriculum Construction

Guangquan Zhang
School of Computer Science and Technology
Soochow University
Suzhou , China
gqzhang@suda.edu.cn

Mei Rong
Shenzhen Tourism College
Jinan University
Shenzhen, China
rongmei@sz.jnu.edu.cn

***Abstract*—"Software Architecture" and "Formal Methods" are two important core courses of Software Engineering. Their teaching quality and effectiveness will have an important influence on cultivating Software Engineering talents with high quality and innovation. At present, the teaching quality and effectiveness of these two undergraduate courses are unsatisfactory and have more problems in the domestic universities. Against these problems, this paper explores in the aspects of teaching content, teaching models, teaching methods and means, teacher resources and so on, in order to improve teaching quality, strive high-level elaborate courses and train qualified software engineering high level talents.**

***Index Terms*—software engineering, software architecture, formal methods, curriculum construction**

I. INTRODUCTION

Software plays an important role in the present information society. Software industry is one of the information society's pillar industries. With the increasingly wide spreading of software applications and the increasingly growing of software scale, people must use engineering approach to develop and maintain software in order to solve software problems efficiently. Software Engineering has become a new discipline from one of computer science subjects.

Internationally, IEEE-CS made software engineering course in the late 1970s.Software engineering education get the support of CMU/SEI in the late 1980s and early 1990s. In 1993, IEEE-CS and ACM founded Joint Steering Committee and Software Engineering Coordination Committee (SWECC). In August 2004, more than 500 professors from universities, research institutions and business all over the world detruded two final version of Software Engineering Body of Knowledge (SWEBOK) and Software Engineering Education Body of Knowledge (SEEK), which marked that software engineering discipline had been established formally in the world and developed rapidly in undergraduate education.

Interiorly, software engineering research began in the early 1980s.Then undergraduate began to set up software engineering course. Software engineering major was set up as the first pilot in 1988. In December 2005, Ministry of Education established Guidance Software Engineering Teaching Sub-Committee to guide the software engineering education of our country. In 2005, the establishment of Chinese software engineering tutorial and undergraduate software engineering specification marked the software engineering teaching plan of our country started entirely.

"Software Architecture" and "Formal Methods", as two important courses of Software Engineering, was carried out among undergraduate of top universities abroad in the mid 1990s.Because the software engineering major is set up late in domestic universities, the core curriculum is just in the initial stage. In the teaching, the phenomenon of difficult teaching and study exists generally and the teaching quality is difficult to achieve the desired objectives. At present, the education of these two courses in domestic universities is mainly faced with the following several problems.

A. *Lack of Application Environment*

Software Engineering is the main application field of "Software Architecture" and "Formal Methods". Domestic software enterprises have made great progress these years while considerable ones still stay at Personal Hero's time and seldom need team cooperation. Under the circumstances, Software Architecture or Formal Methods can't get real promotion, which cause that some large enterprises have to introduce talents from foreign country.

B. *Scarcity of Curriculum Materials*

At present, there are only several teaching materials about "Software Architecture" and "Formal Methods" in our country. Formal Methods basically uses some method, such as Z, B and Petri Nets and so on. The software engineering or hardware design teaching materials about Formal Methods are nothing but two or there kinds, which greatly restrict the teachers' comprehensive grasp about Formal Methods.

C. *Lack of Support tools*

In the education of "Software Architecture" and "Formal Methods", the relevant support tools are indispensable in order to improve learning interest and teaching effectiveness. However, it is difficult to find valuable support tools. The primary cause is that from understanding to using tools need a

978-1-4244-9718-8/11/$26.00 ©2011 IEEE

lot of energy. It also needs more energy to develop the corresponding teaching support tools.

D. Lack of Experimental Environment

"Software Architectures" and "Formal Methods" are very important for developing large-scale systems. Usually, schools can only provide small-scale teaching experimental environments which are far behind the large ones. And the students do not have the opportunity to practice in large companies. They cannot form a perceptual sense for formalism.

To settle problems mentioned above, it is necessary to carry out research on "Software Architectures" and "Formal Methods" which are two core courses of software engineering. Combined with CDIO mode, the latest achievement in international engineering education, we carry out reforms on teaching contents, models, methods and tools.

This paper focuses on the shortcomings of current teaching quality and effects of undergraduate courses, "Software Architectures" and "Formal Methods" of software engineering program.

II. Exploration and Practice in "Software Architectures" Curriculum Construction

Software Architectures are subjects developed from the practice in software engineering. Having developed for 20 years, it has become the core of development of complex software systems. Software Architectures focus on the high-level architecture of complex software systems, describe relations between organization units and carry out activities based on the description (for instance, designation, assessment, realization, management and test). They contain various aspects and have high-degree of abstraction. Due to these, it is hard for students who have just got in touch with software engineering to understand.

A. CDIO Model Based Software Architecture Case Teaching Method

In consideration of the course's high-degree of abstraction and students' lack of experience, it is hard to achieve good effects by teaching according to the teaching materials. So we propose a new software architecture teaching method based on CDIO model case study. In that case, we can combine abstract theories with specific cases. The cases selected should obey the following rules: first, cases should be chosen from practical application systems. Usually, the application system is very large. We carry out extraction of the original system to suit classroom instruction. Once the students master the skills, they can apply them into operation, which can improve their software development capabilities. Second, cases should have close connection with teaching contents and current software situations. This course selects cases according to software architecture design mode, style and current Software Architectures. Third, cases can be extended. We leave interfaces in each case study and require students to extend the cases in consideration of knowledge from other courses. This is not only good for students mastering the knowledge, but also good for training students' ability of doing things by themselves.

B. Reform the Teaching Contents, Deliver Courses Based on Special Topic Research

Due to the wide range of Software Architectures, the advanced and unpractical of current teaching materials, we carry out reform on teaching contents. Following the clues of special topic research, we can make clear about contains of sub-fields of Software Architectures and relationships between sub-fields. Students can be divided into groups to find specific cases according to their own interests. At the same time, we use extra points to encourage some students with work experience. They can start from tackling the practical problems and complete the solutions to the questions during the learning process.

C. Bring in Double-Teacher Mechanism, Establish Diversified Teaching Staff

This course requires for high-quality of teacher's knowledge structure and engineering background. Considering the shortage of qualified teachers, we bring in double-teacher mechanism. That is, apart from the college teachers, we bring in software experts to deliver lectures and reports. In this way, we can establish a diversified teaching staff.

III. Exploration and Practice in "Formal Methods" Curriculum Construction

Formal Methods are rigorous, mathematical and logical methods which can be used to perform research on computer systems. "Formal Methods of software engineering" is listed as a core course by ACM and IEEE-CS's CC2005 software engineering SE2004. European Formal Methods association established a branch FME-SoE about Formal Methods education in 2001. And it published reports on Formal Methods education courses for 58 colleges and universities in 11 European countries. All the work has a positive effect on improving Formal Methods education in software engineering program.

Formal Methods embodies the necessity and feasibility increasingly because of needed in the industry application and experience accumulation in teaching practice. But Formal Methods teaching was valued and popularized by Occident University and Colleges just in last ten years. It still needs to build perfect knowledge hierarchy and teaching content. At present, domestic Formal Methods teaching in the software engineering is weak. It isn't popularized and enforced effetely. This research put forward a teaching method: professional language/method teaching, comprehensive content teaching and scatter knowledge point teaching.

A. Professional Language/Method Teaching

Just teach some special or professional language/method, such as Z, B, and VDM, Petri nets, logic and proof procedures. This way can apply to lower grade student. Its purpose is in order to train the student's ability inosculating the Formal Methods into other courses. And train the student's ability using Formal Methods in software and hardware designing step by step. But the limitation of this way is that it just contains the single method/language, so the student can't know the Formal Methods deeply and extensively. And in the

meantime, by the variety of the Formal Methods used by commercial software, it's hard to guarantee the student can use the Formal Methods. The deviation between teaching and practice is large.

B. Comprehensive Content Teaching

Teach multiple Formal Methods/languages around one theme or knowledge unit. For example, Formal Methods in software developing, formal specification, formal verification, protocol engineering and so on. The advantage of this mode is that it doesn't trap the student into the thought just learning some Formal Methods. Meanwhile it also can let the student learn the methods useful for him to apply in practical software developing. Due to the extensively coverage of the course, less deep of the basic theory and principle, it's easily to mislead student. Let them think that Formal Methods is just a kind of symbol method to know things. And let the student hard to combine the software developing with Formal Methods.

C. Scatter Knowledge Point Teaching

Embed the knowledge point of Formal Methods into the course teaching, such as software architecture, software test, software requirement engineering. The advantage is applying the Formal Methods throughout the software course teaching following in proper sequence. It develops the student's ability thinking in Formal Methods and developing software by Formal Methods. This mode has a challenge that it's need to revise all software engineering course content. And it's also a challenge to teachers.

IV. CONCLUSION

System architectural and analyst are chief designers in a project. They are the constructors of new products and new technology system and the high level talents needed in software project.

Software Architecture and Formal Methods have the important position in modern software industry. They are the important courses to train the qualified architectural and analyst. However, it's hard to achieve the goal of teaching because of the abstractness and complexity. For this reason, this subject research innovates and improves some aspects such as teaching content, teaching method and son on. It improves the teaching quality by combining the abstract theory with practical application. Finally, we hope to turn out the high level talents to promote our country's software engineering and large complex systems progress.

For training our country's high level talents-software architectural and analyst, this article has the important meaning.

ACKNOWLEDGMENT

This work is partially supported by Computer Teaching Reform Key Project of Soochow University (2010); Nation University Students Innovative Pilot Scheme (Soochow University,No:101028524);Open Funds of the StateKey Labor -atory of Computer Science of Chinese Academy of Science (SYSKF0908); Natural Science Foundation of the Jiangsu Hig -her Education Institutions of China(08KJB520010).

REFERENCES

[1] Zhang Guangquan. Discussion and practice of school-enterprise cooperation oriented software engineering talents training mode, Computer Education, 2008.21,29-32(in Chinese)

[2] Zhang Guangquan. Exploration and practice of multi-level software engineering talents training, Computer Education, 2005.12,40-41(in Chinese)

[3] Rong Mei, Zhang Guangquan. Discussion of practical software engineering talents training methods, The third college computer courses report forum proceedings, High Education Press,2008.6:311-314(in Chinese)

[4] Rong Mei, Zhang Guangquan. Strengthen the practical innovation ability and Breakthrough the software talent bottleneck ,National computer teaching experiment conference, Research and Exploration in Laboratory, 2007,26(12):198-201(in Chinese)

[5] Zhang Guangquan. Reflections about improving the quality of graduate education, National Computer Education Conference 2006, Computer Education,2006.8:86-88(in Chinese)

[6] Zhang Guangquan.93 teaching plan and construction of the 21st century computer disciplines, National Computer Education Conference 1999,Computer Science,1999,26(no.10 special):161-163(in Chinese)