CHAPTER 10: SUMMARY AND FUTURE SCOPE OF WORK

Summary
In this study, we have proposed a three-tier taxonomy of twelve competencies for software development education. It includes five basic competencies, three ‘competency driver-habits of mind,’ and four ‘competency conditioning attitudes and perspectives.’ The five basic competencies are: (i) technical competence, (ii) communication competence, (iii) domain competence, (iv) complex problem solving competence, and (v) computational thinking competence. The three ‘competency driver-habits of mind’ are: (i) attention to details, (ii) critical and reflective thinking, and (iii) creativity and innovation. The ‘competency conditioning attitudes and perspectives’ include: (i) intrinsic motivation to create/improve artifacts, (ii) curiosity, (iii) decision making perspective, and (iv) systems-level perspective.

We have reviewed the educational research literature to examine its applicability for developing these competencies through appropriate interventions for instructional reform. We have done many empirical (qualitative and quantitative) studies among students, faculty, and professionals, to find out the preferred approaches of learning and effective pedagogical techniques. Our empirical studies suggest that didactic approaches of teaching are ineffective. Students experience much deeper learning in active, integrative, reflective, and collaborative constructive environment.

Hence, we have proposed a comprehensive unified framework of pedagogic engagements. Our proposed framework of pedagogic engagements in software development education is grounded in (a) core activities of software development, and (b) distinguishing characteristics of software development profession. It includes - (i) three-tier taxonomy of twelve core competencies, (ii) five-dimensional ladder of professional and human development, (iii) three-dimensional perspective of the knowledge domain of software development, (iv) two core principles (cognitive dissonance and cognitive flexibility) for facilitating deep learning, and (v) a four-dimensional taxonomy of pedagogic engagements (active, integrative, reflective, and collaborative) over (iii) for developing (i) and (ii).
We have also discussed some instructional interventions developed by us, manifesting some aspects of our framework. These interventions were administered in a chosen set of existing computing courses. Some new courses have also been developed in the process. The development of the framework of pedagogic engagement, and these interventions for instructional reform of software development education, has been an intertwined and highly spiral process. Large classes offer a huge challenge. There is a need to explore the possibility of a complete revamp of the software development education and curriculum through our framework. While some interventions have been successfully tested with large classes, others were not as successful for large numbers. For example, the use of inquiry teaching in lecture classes offers huge benefits to learning oriented students, it has not been found to be as attractive to exam oriented students.

**Future Scope of Work**

We have discussed our experience in conducting reflective workshops on pedagogy for engineering faculty. More work needs to done in designing teachers’ training programs on pedagogy. We intend to use our framework to design many such workshops to motivate the teachers to use aspects of our framework in their teaching [402].

We hope that our proposed framework of pedagogic engagement in software development education will help the community of software development educators and researchers to create a variety of interventions that will help in extending the ‘Software Engineering Body of Knowledge’ (SWEBOK) to ‘Software Development Education Body of Knowledge’ (SDEBOK).

The curriculum, syllabus, and textbooks often ignore many professional as well as pedagogical aspects. Our proposed framework of pedagogic engagements of software development education, offers the potential to redesign the instructional material for all computing courses. Systematic projects can be initiated in this direction.
Reflection has been found to be a highly effective pedagogical engagement. However, its use in computing courses is not very popular. Future work is required to systematically incorporate this aspect in student assignments in all computing courses and projects.

Systems-level perspective is one of most important competencies for software developers. The development of system-level perspective depends upon students’ engagement with a curriculum and courses that are themselves designed with this perspective. The curriculum as well all courses need be redesigned as systems, where not only the computing course, but also the other courses, offered by other departments for computing students, will also be well integrated into a single whole. Our approach of multi-level infusion offers a way out. This will also help in increasing domain sensitivity and expertise of computing students.

Project centric evolutionary teaching offers active, integrative, reflective, as well as collaborative engagements as per our framework. Future work is required for using this approach in different computing courses.

A novel approach of collaborative pair and quadruple programming has been proposed. The results of sample tests were found to very encouraging. Further work is required to examine the impact, and investigate ways of pervasively integrating it into all computing courses. More research is required to create different types of collaboration models in the context of different computing courses and projects.

Multi-level infusion opens a new way of transforming the computing courses. In collaboration with various faculty members, we continue to strengthen the infusion of eight elements: web technology, multimedia technology, mobile technology, security aspects, systems design aspects, estimation aspects, open source, and debugging in various introductory core computing courses. This is bringing deeper integrated learning, higher levels of enthusiasm, and challenge in the courses. We have also started working towards designing appropriate models for multi-level infusion of selected elements of software documentation, software quality, software risks managements, advanced level programming techniques, and formal methods of software
engineering. More work is required to develop detailed instructional material using this approach.

*Cross-level mentoring* has been found to highly effective wholesome engagement for senior students. More work is required to integrate this approach within the educational systems. Many new ways of forging collaborations between senior and junior level students need to be invented to create a collaborative community of co-learners.

We also believe that the proposed framework and our research approach are fairly comprehensive, reusable, and robust. Designers of *educational programs for other professions* can also adapt this framework and methodology.

More research is needed in developing new models and exemplars for offering multi-dimensional engagement to the users of online education and e-learning programs [403-408]. *Our framework of pedagogic engagements can be suitably adapted to create a framework of pedagogic engagements in e-learning and online environments.*