Chapter 74
Teaching Software Architecture in Industrial and Academic Contexts: Similarities and Differences

Paolo Ciancarini
University of Bologna, Italy

Stefano Russo
University of Naples Federico II, Italy

ABSTRACT

In this chapter, the authors describe their experiences in designing, developing, and teaching a course on Software Architecture that tested both in an academic context with their graduate Computer Science students and in an advanced context of professional updating and training with scores of system engineers in a number of different companies. The course has been taught in several editions in the last five years. The authors describe its rationale, the way in which they teach it differently in academia and in industry, and how they evaluate the students' learning in the different contexts. Finally, the authors discuss the lessons learnt and describe how this experience is inspiring for the future of this course.

INTRODUCTION

What is the role of the software architecture inside a large mission-critical system? How is it created? How is it managed? How can the concept foster software reuse and productivity? These questions are quite relevant for engineering companies, which produce families of software intensive systems (Buschmann, 1996). As software systems become larger, more complex, and more expensive, companies - in particular system integrators - feel an increasing need for improving their productivity exploiting sound and effective techniques for the definition, analysis, and evaluation of software architectures. This is what we observed in a number of cooperations between academia and industry, and that motivated our study of how Software Architecture can be taught.

DOI: 10.4018/978-1-5225-3923-0.ch074

The initial question from which we started our study was: "how can one introduce and teach software asset reuse and software architecture evaluation to engineers who have been designing systems for years without explicitly dealing with these concepts?" Then we added a related question: "how can our experience in teaching Software Architecture in an industrial context be imported in an academic context of Computer Science students?"

When the field of Software Architecture emerged, it was argued it should be considered a discipline in its own, separate from Computer Science and possibly encompassing Software Engineering (Clements, 2010). After almost twenty years the corpus of the scientific works in the field has developed consistently, but there is still a large gap between this body of knowledge and what is actually needed in academic and industrial settings. Many of the achievements in the field have not matured enough; an example are Architecture Definition Languages (ADLs), that have not replaced - and are not likely anymore to replace - standard modeling languages (Clements, 2012). Some others achievements are more mature, for instance some tools for architectural analysis: in (Bernardo, 2001) is described a tool for performance evaluation of a software architecture, whereas in (Sterling, 1996) is described a tool for architecture animations. However, these tools still need to be tailored to specific software systems, and even more to internal industrial practices.

The increase of the size and complexity of contemporary software-intensive systems raises critical challenges to the engineering companies which build them to be integrated into larger systems of systems. Production and management problems with software intensive systems are well known and related to requirements engineering, software design, systems' families management, and their continuous testing integration and evolution.

Thus, teaching software architectures in an industrial context requires to meet a company's expectations in terms of mature knowledge, special competences, and best practices transferred to practitioners, that they can subsequently turn into the engineering life cycle of complex systems.

This is not easy to achieve, as architecting large-scale complex software systems - having tens of thousands of requirements and millions of lines of code - requires very high abstraction and modeling skills. A number of methods and solutions to these problems are based on the introduction of software architecting in the industrial practice (Bass, 2012). However, to become effective an architecture-centric development approach needs to be assimilated and put in everyday practice by the company personnel, who need architectural education and training.

We have found that introducing Software Architecture principles, methods, and tools in an academic context poses different problems because Computer Science students are less expert and more interested in creativity and technology use rather than software reuse and architectural evaluations. Thus, in our classes we emphasize the relationship of Software Architecture with programming languages and formal methods for modeling and reasoning on software systems.

In this chapter we describe our experience in designing, developing, and teaching a course on Software Architecture, that we tested both with our graduate Computer Science students in an academic context and with several systems engineers in a number of different companies, in a context of professional updating and training. The course has been taught in several editions in the last five years. We describe its rationale, the way in which we teach it differently in academia and in industry, and how we evaluate the students in the different contexts. Finally, we discuss the lessons we learnt and describe how this experience is inspiring us for the future of this course.