Chapter 8.3
Modern Approaches to Software Engineering in the Compositional Era

Ali Dogru
Middle Eastern Technical University, Turkey

Pinar Senkul
Middle Eastern Technical University, Turkey

Ozgur Kaya
Middle Eastern Technical University, Turkey

ABSTRACT

The amazing evolution fuelled by the introduction of the computational element has already changed our lives and continues to do so. Initially, the fast advancement in hardware partially enabled an appreciation for software potency. This meant that engineers had to have a better command over this field that was crucial in the solution of current and future problems and requirements. However, software development has been reported as not adequate, or mature enough. Intelligence can help closing this gap. This chapter introduces the historical and modern aspects of software engineering within the artificial intelligence perspective. Also an illustrative example is included that demonstrates a rule-based approach for the development of fault management systems.

INTRODUCTION

Since its earlier days the audience was attracted to the notion of 'electronic brain' that potentially could aid, replace, and even improve human-level intelligence. Expectations from such artificial intelligence, in improving the daily life, now had to be extended to aid the very techniques that

DOI: 10.4018/978-1-60960-818-7.ch8.3

would develop the artificial intelligence. Software Engineering could definitely benefit from such leverage while software demand has grown exponentially as ever and corresponding offer – i.e. software development, could never catch up with. Fortunately, before the end of its half a century long quest, after experimenting with improvised notions the inevitable incorporation of intelligence is in the process of being established.

It is desirable to replace the human in the software process, as much as possible. Linear improvements through better techniques but still targeting the addition of the next single block at a time, to build a huge system, are not sufficient. The development technique needs to be automated and automation is an open invitation to Artificial Intelligence (AI). Software engineers are under pressure for addressing organized ways for utilizing new approaches such as Component Orientation, Service Orientation, Aspect Orientation, Software Product Line Engineering, Model Driven Development, and related approaches. If not due to a conscious planning that is aware of the necessity and the opportunities, this consideration is at least being motivated by such technologies getting more and more popular. Such approaches assume the role of "enabling technologies" for the "Compositional Era". Today's software engineer should expect to deliver a system through locating and integrating sub-solutions in contrast to creating them. AI will help the engineer in automating the search and integration of the sub-solutions. Consequently, our methodologies that are aligned towards defining the code to be written and eventually writing it one line at a time, are changing towards locating and integrating blocks. AI will help the paradigm shift from code writing to composition.

Our History in the AI Perspective

Manifesting itself after the introduction of the modern computer, the older dated field of Computer Science (CS) surfaced with the goal to reduce the "semantic gap". This was simply to convert problem representations from the conventional ways we developed for human understanding, to another, that computers could understand (execute). Figure 1 depicts the semantic gap concept.

This clearly states the main issue as that of Knowledge Representation (KR). KR in return, is a fundamental AI notion. In other words, the computation field is basically trying to enact AI. Among the many sub-fields only these that concern the higher-level regions of the semantic gap are somehow regarded as related to AI. One lowest level tool, covering the bottom region of the gap is the compiler. Neither a compiler, nor an embedded software that drives a dishwasher for example, are considered as artificially intelligent.

Figure 1. Semantic gap