Chapter 17

Building the IT Workforce of the Future: The Demand for More Complex, Abstract, and Strategic Knowledge

Deborah J. Armstrong
Florida State University, USA

H. James Nelson
Southern Illinois University Carbondale, USA

Kay M. Nelson
Southern Illinois University Carbondale, USA

V.K. Narayanan
Drexel University, USA

ABSTRACT

The software development process has undergone a considerable amount of change from the early days of spaghetti code to the present state of the art of development using strategic patterns. This has caused not only changes in the toolkits that developers use, but also a change in their mindset—the way that they approach and think about software development. This study uses revealed causal mapping techniques to examine the change in mindset that occurs across the procedural to OO development transition, and lays the foundation for future studies of the OO/ pattern cognitive transition. The results indicate that there is not only increasing complexity in the cognitive maps of the OO developers, but also that there is a need for the developer to shift from routine, assembly line coding to more abstract thought processes.
INTRODUCTION

No one doubts that the software development process has undergone a profound transformation. Twenty years ago, the state of the art was the waterfall model of the systems development life cycle. The project planning and feasibility study steps were followed by systems analysis and requirements gathering, system design, coding, integration and testing, and finally installation and maintenance. The waterfall model fit very nicely within the rigid hierarchical organizational structures of the time. Functional silos and economies of scale drove software development. Systems analysts created data flow diagrams and ER diagrams and passed these to the designers. Designers would create functional decomposition diagrams and relational data models and pass these to the coders. Finally, the coders rendered all these into COBOL, FORTRAN, or a number of other procedural programming languages and database management systems. The constant translation from model to model enforced a sequence on the development process (Coad & Yourdon, 1991), with the side effect of keeping each different kind of developer in his or her place. Expert coders could not easily transition to the more abstract world of analysis and design, while coders had to move to the more abstract world of analysis and design, designers had to become more analytical. Analysts, in turn, needed to move from relatively limited requirements analysis to the much more extensive domain analysis and the development of organizational information architectures (Eversden, 1996). Further, the breakdown of the barriers between the analyst and the designer and the use of models that span the lifecycle (for example, UML, Booch, Rumbaugh, & Jacobson, 1999) has created the need for more extensive project management skills. The shift from “doing analysis” then “doing design” structured by the different models in use has been replaced by an iterative and seamless development life cycle. However, just as coders have difficulty transitioning to the more abstract world of analysis, expert analysts have difficulty transitioning to the strategic world of the project planner (Nelson & Nelson, 2003).

This trend is continuing with the beginning of another software development revolution. One of the causes of recent technology project failures is the disconnect between organizational strategy and technology (Lufman & Brier, 1999; Lufman, 1996). An example of this disconnect can be found in the implementation of enterprise resource planning (ERP) systems. ERPs were sold as strategic enterprise solutions, even though at their core is a set of integrated, somewhat standardized business processes (Lee, Siau, & Hong, 2003). This approach to product development had virtually no strategic intent, and therefore ERPs, while sometimes solving process level problems,