IRM PRESS
701 E. Chocolate Avenue, Suite 200, Hershey PA 17033-1240, USA
Tel: 717/533-8845; Fax 717/533-8661; URL-http://www.irm-press.com

ITB11775

This chapter appears in the book, *Skills for Managing Rapidly Changing IT Projects* by Fabrizio Fioravanti © 2006, Idea Group Inc.

Chapter VII

The Target Business Environment

During the last few years, whoever has been involved in software development and management has noticed that two different kind of projects have arisen: very large projects carried on by large companies over long periods of time (i.e., 1 to 3 years) and small to medium projects developed in a shorter period (generally less than 1 year) by small companies.

The second type of project deals mainly with recent technologies; these projects are usually Internet-related and are subject to change often and deeply in the specification (Boehm, 2000; Cusumano, 1999; Kroeker, 1999; Miranda, 2002). Moreover, since the latest technologies are on the edge and change quickly in their specifications, the technologies adopted during the project lifetime are subject to modification, and therefore, the code usually has to deal with deep modifications. The second part of this book is focused mainly on this second type of project that is often described as *rapidly changing*. In the following chapters, I will offer suggestions on how to manage and develop such projects in an environment that continuously changes and how to adapt to user modified needs and deal with technologies drift.

Several books and papers have been written with the aim of addressing the problems that arise from large projects. They teach you how to manage and develop where requirements are clear, where specification is done in advance,

and where the code written today is reused without changing after 1 year. Obviously, this is still true for several large projects, especially related to banks, large industry, production companies, and similar environments where the customer knows exactly what he or she wants and is capable of becoming stable during the full life cycle of the project, the technologies, the underlying hardware, and, generally speaking, the requirements (at least with an approximation of 95% to 98% for each year).

Unfortunately, the real world of software engineers, managers, and developers is full of small companies that have customers that can draw specifications only after they have seen the first prototype, which generally does not satisfy them. In general, the IKIWISI (*I'll know it when I see it*) (Boehm, 2000) approach guides the customer, compelling deep modification of the structure, the requirements, and the code that are part of the project.

In order to avoid or at least to mitigate the effects of this refactoring on a classical management methodology such as waterfall, spiral, and so forth, Agile methods for software development and management have been created. More recently, Agile methodologies (Beck, 1999, 2000; Boehm, 2002) are trying to partially abandon their pure concepts in order to approach the world of more formal methods, such as Capability Maturity Model (Paulk, 1993, 1993a, 2001), UML (Booch, 1998; Favre, 2003; Larman, 2001; Mellor, 2002; Rumbaugh, 1999), and tools for aiding the formalization of concepts and for automating the generation of documentation.

In information technology, an amazing solution does not exist for developing and managing projects, since a single development and management methodology is not suitable for all companies, for all projects, and last, but not least, for each developer and manager.

Moreover, software engineering has no silver bullet to solve the problems related to each different type of project, and therefore, it is necessary to define the target environment for which the methodologies and techniques reported in this book are valid and can be successfully applied.

The book title refers in a generic way to rapidly changing projects. Which projects can be defined as rapidly changing? A rapidly changing project is not a rapid project, since some long projects or projects where constraints and bounds changes rapidly also can have these characteristics.

Generally speaking, if several of the assumptions that follow apply to your project, you can be quite sure that you have a rapidly changing project. You have to consider the following list as a set of rules of thumb; the reader can