Chapter 10
The Anatomy–Centric Approach Towards Managing Complex Projects

The purpose of this chapter is to describe and discuss the anatomy-centric approach towards coordinating complex development projects. The content of the chapter is mainly based on my empirical experience from Ericsson.

The development of a telecom system must deal with a multitude of technical, market related, cultural and organizational interdependencies. Most often, this requires mutual adjustment across many types of boundaries (Adler, 1999). For example, telecom systems are to a great extent implemented in software and developed on a global basis. The interest in distributed software development has increased due to factors such as reduced costs, the access to well-educated labor pools, the possibility of 24-hour development, global presence, and proximity to customers (e.g. Battin, Crocker, Kreidler, & Subramanian, 2001; Damian, 2003; Ebert & De Neve, 2001; Gorton & Motwani, 1996; Heeks, Krishna, Nicholson, & Sahay, 2001; Herbsleb & Moitra, 2001; Karlsson, Andersson, & Leion, 2000; Komi-Sirviö & Tihinen, 2005; Kraut & Streeter, 1995; McChesney & Gallagher, 2004; Ovaska, Rossi, & Marttiin, 2003; Prikladnicki, Audy, & Evaristo, 2003; Sakthivel, 2005).

Many challenges already present in centralized software development are aggravated by the distribution. At the core of these challenges lies the issue of coordination. “While there is no single cause of the software crisis, a major contributor is the problem of coordinating activities while developing large software systems” (Kraut & Streeter, 1995, p. 69). The anatomy-centric approach is one possible answer to these challenges.

DOI: 10.4018/978-1-60566-192-6.ch010
FROM WATERFALL TO INCREMENTS

During the late 1980s and early 1990s many software producing organizations were trying to find alternatives to the then prevalent “waterfall” model for software development (e.g. Brooks, 1995). Examples of such alternatives are evolutionary delivery models, spiral models, iterative development, etc. The traditional waterfall model consisted of phases performed in sequence: analysis, implementation, and integration & test. In the analysis phase, the total functionality of the system was distributed to subsystems and modules, which were developed and tested in the implementation phase. In the integration phase, modules were gradually integrated and system tested until the entire system functionality was achieved.

However, the waterfall model had several intrinsic problems (ERI-1996-04-04; Karlsson, 2002):

- The time span between specification of the system and feedback to customers was wide. After providing the requirements at the beginning of the project, the customer had to wait for a long time before seeing any results.
- Requirement changes were hard and expensive to accommodate for. Changes in functionality might lead to the re-opening and re-design of already completed modules.
- The delayed integration of modules (called “Big Bang” integration) meant that potential interface problems between the modules were discovered late in the project. Moreover, the concentration on testing and integration towards the end of the project put a heavy burden on test personnel and system test equipments.
- The early distribution of the total functionality on modules brought about a focus on module quality rather than system quality. There was a tendency for designers to concentrate on their specific module, and paying less attention to interactions between modules.
- The focus on modules resulted in an impaired project control. Since the progress of the project was measured from how many modules were ready, problems caused by potential integration problems and/or lack of customer focus appeared late in the project, which might have caused unforeseen delays.
- Since the project participants executed each phase only once, the potential for learning was low. Good and bad experiences collected in one phase were hard to apply anew as the first opportunity to do so did not come until a new project was started. By that time, most lessons learnt had been forgotten.

These and other problems stimulated Ericsson to look for alternative development models. Various more or less isolated initiatives were taken at different development units during the early 1990s. All these initiatives had in common some kind of incremental development approach, where the total functionality of the system was built in several steps in such a way that each step provided a testable part of the overall system. This is illustrated in Figure 1.

Several advantages were anticipated from the transition from the waterfall to the incremental model:

- Since the increments can be developed to a certain extent in parallel, the lead-time of the project can be reduced. Although each phase in fact stretches over a longer period of time (see Figure 1), the overall time span is shorter. However, this reduction comes at the expense of a more
Related Content

Parametric Generator for Architectural and Urban 3D Objects
[www.igi-global.com/chapter/parametric-generator-architectural-urban-objects/69466?camid=4v1a](www.igi-global.com/chapter/parametric-generator-architectural-urban-objects/69466?camid=4v1a)

Iterative Fault Tolerant Control for General Discrete-Time Stochastic Systems Using Output Probability Density Estimation
[www.igi-global.com/chapter/iterative-fault-tolerant-control-general/69670?camid=4v1a](www.igi-global.com/chapter/iterative-fault-tolerant-control-general/69670?camid=4v1a)

Visualization of Insight Process in Concept Creation Focusing Handwriting Features
[www.igi-global.com/article/visualization-insight-process-concept-creation/76328?camid=4v1a](www.igi-global.com/article/visualization-insight-process-concept-creation/76328?camid=4v1a)

Processing Combinatorial Thinking: Innovators Marketplace as Role-Based Game Plus Action Planning
[www.igi-global.com/article/processing-combinatorial-thinking/101153?camid=4v1a](www.igi-global.com/article/processing-combinatorial-thinking/101153?camid=4v1a)