5.1 INTRODUCTION

Although the situation in the software industry is improved in the last years, the percentage of software project cancelled 18%, or challenged (late, over budget, and with less than the required features) 53% is still high. Researchers and practitioners are looking for the magic solution or the silver bullet that will allow software companies to overcome the software crisis (Brooks, 1987). New development approaches like AMs and OSD models are some of the solutions identified (Feller & Fitzgerald, 2002; Abrahamsson et al., 2003).

One critical problem in software development consist of coordinating interdependent processes involving many interacting stakeholders with different interests, points of view, and expectations (Toffolon & Dakhli, 2000).

Two inherent characteristics of software make coordination a critical issue and a primary cause of the software crisis. These characteristics are (Kraut & Streeter, 1995; Marchesi et al., 2002):

1. **Complexity**: software systems are often very large and far beyond the ability of any individual or small group to create or even to understand in detail. Several, different groups work toward a common goal, share information, and
need to coordinate their activities. A lot of interdependent decisions have to be made in the presence of incomplete information and high uncertainty.

2. **Uncertainty**: software development is permeated by uncertainty. Environmental factors like technological innovations and market changes introduce uncertainty into software projects. The intangibility of software makes it difficult to review progress, thus introducing uncertainty into software project management. The flexibility of software and the volatile environment that software projects face makes change a main source of uncertainty. Changing and incomplete requirements increase uncertainty in software development. Finally, software is uncertain because the different subgroups involved in its development often have different beliefs about what it should do and how it should do it.

Coordination becomes much more difficult as project complexity and uncertainty increase (Kraut & Streeter, 1995).

This chapter focuses on how to coordinate software development when there is a high level of complexity and uncertainty. Companies adopting new development approaches like AMs and OSD models are investigated. In fact, AMs are suitable for developing software in dynamic and uncertain environment, while OSD models develop software in particularly complex conditions.

Using the framework provided by organization and game theory, we are going to identify the main kind of dependencies and coordination mechanisms both in OSD models and AMs.

### 5.2 WHAT IS COORDINATION?

#### 5.2.1 Organization Theory

A common way to deal with complex problems consists of splitting the problem in smaller sub-parts easier to manage. The Fordist approach in product manufacturing is an example of this method (Ohno, 1988). According to the Fordist approach, division and specialization of labor permit to achieve perfection in the tasks and economies of scales (Womack *et al.*, 1991). In particular, complexity can be tamed through:

- Clear identification of different activities to do
- Specialization of competences

Clear division of tasks and separation of labor reduces complexity but can increase the integration need of the organization. Specialization causes *interdependency*
Teaching Software Engineering with Free/Libre Open Source Projects
www.igi-global.com/article/teaching-software-engineering-free-libre/2772?camid=4v1a