8 Architected Agile Solutions for Software-Reliant Systems

Barry Boehm, Jo Ann Lane, Supannika Koolmanojwong, Richard Turner

Abstract: Systems are becoming increasingly reliant on software due to needs for rapid fielding of “70% capabilities,” interoperability, net-centricity, and rapid adaptation to change. The latter need has led to increased interest in agile methods of software development, in which teams rely on shared tacit interpersonal knowledge rather than explicit documented knowledge. However, such systems often need to be scaled up to higher level of performance and assurance, requiring stronger architectural support. Several organizations have recently transformed themselves by developing successful combinations of agility and architecture that can scale to projects of up to 100 personnel. This chapter identifies a set of key principles for such architected agile solutions for software-reliant systems, provides guidance for how much architecting is enough, and illustrates the key principles with several case studies.

8.1 Introduction

Systems are becoming increasingly reliant on software due to needs for rapid fielding of “70% capabilities,” interoperability, net-centricity, and rapid adaptation to change. This trend is shown in Figure 8.1, illustrating the percentage of aircraft functionality that relies on software versus time, and the resulting system challenges as software was used to adapt to change (Van Tilborg, 2006).

In (Boehm and Turner 2004), we identified two fundamental approaches to developing software systems: agile and plan-driven. Given that there is a broad spectrum between these two extremes, we characterized “home grounds” that describe the sets of conditions under which each is most likely to succeed. The more a particular project’s conditions differ from the home ground conditions, the more risk there is in using one approach in its pure form, and the more valuable it is to blend in some of the complementary practices from the opposite method. Table 8.1 summarizes these home grounds.

Overall, small, less mission-critical projects with high rates of change are best accomplished by highly skilled agile teams able to operate on shared, tacit knowledge and little documentation. Large, more mission-critical projects with less change and mixed developer skill levels are more successful using explicit documented knowledge, such as architectural views and project plans, to succeed. If the requirements of these large projects are relatively stable, architectures and
plans will change infrequently, so a pure documented-architecture approach will succeed. However, as seen in Figure 8.1, large projects increasingly need product and process architectures that enable them to use agility to support more volatile requirements in areas like user interfaces, competition-critical features, or interfaces with independent, rapidly evolving external systems.

Also, our experiences in the commercial and public-service sectors indicate that a growing number of systems must integrate into larger enterprise software frameworks and systems of systems. This leads to complex interactions with evolving COTS products, legacy systems, and external systems, with the tacit expectation that these systems “never fail.” Emergent requirements, rapid changes, reused components, high levels of assurance, and dynamic market factors further complicate integration or incorporation.

**Fig. 8.1.** Trends in Software-Reliant Systems: Aircraft

The relative importance of these different application sectors depends on what is counted. Organizations dealing mainly with large numbers of small, dynamic, less-critical projects will count the number of such projects. Organizations dealing with a mix of large and small projects will more likely count the investment costs and skilled effort that are consumed by their projects.

Figure 8.2 shows how the relative importance of agility and architecture varies when counting numbers of projects or percentage of costs. Drawn from Appendix E of (Boehm and Turner 2004), the information is based on data provided in (Highsmith 2002) of the relative number of projects in three size ranges across various business sectors. The particular data shown in Figure 8.2 is from the fi-