

# The Tightrope to E-Business Project Success

AMRIT TIWANA AND EPHRAIM R. MCLEAN

Even though U.S. businesses spend half of their business equipment budgets on IT, only one out of six software projects is successful. Annually, this adds up to about \$75 billion a year in failed projects, \$22 billion in cost overruns, and another \$175 million for systems that deliver less than they claim [9]. The elusivity of success is rooted in organizations' inability to manage—or even recognize—the unexpected. Inattention to the emergent gap between what's delivered and what's finally needed, not technical incompetence, is what stumps even the best of intentions. Code, however good, cannot compensate for poor conceptualization.

A new philosophy that views software as a medium in which knowledge is embedded rather than a product is emerging in some circles of the software community [2, 8]. Proponents of this view encourage embedding accurate knowledge into the design of a system over trying to manage requirements. Software project execution is like walking a tightrope from idea to implementation. It takes just the right balance between methodology and imagination, and following rules and knowing when to break them. Either excess slack or inflexibility guarantee a fall. We conducted a field study to assess the merits of this view and determine whether embedding knowledge in software design influences how well a project adapts to changing technology, unstable business needs, and customers who cannot seem to make up their mind. We found strong support for this emerging perspective: project teams that excel at knowledge integration best manage the unexpected to successfully execute innovation-intensive software projects. This article discusses the implications of our findings for software practice.

## From Engineering Imagination to Imagination Engineering

The most difficult part of building a new system is determining what to build. Determining that is clearly a process of reducing ignorance about customer needs, technology, and a project's business context [3]. A system is the byproduct of such knowledge. When that knowledge itself is incomplete, a project is built on an architecture of misunderstood business needs.

---

AMRIT TIWANA (ATiwana@bus.emory.edu) is an assistant professor of Decision and Information Analysis in the Goizueta Business School at Emory University, Atlanta.

EPHRAIM R. MCLEAN (EMcLean@GSU.edu) is a Regents' Professor and the Smith Eminent Scholar's Chair in Information Systems in the Robinson College of Business at Georgia State University in Atlanta.

---

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

---

Thinking of requirements in terms of code rather than ignorance-reducing knowledge is the seed of many problems. Conventional requirements elicitation processes are a good example. Even if requirements are accurately elicited, there is little guarantee that the answers or the questions asked to determine them will remain unchanged over the course of a project. Ensuring the relevance of knowledge embedded in the design of software requires ongoing integration of fragmented customer and domain knowledge into project-level knowledge, not a static, one-shot specification document. The product view of software fails to absorb high requirements ambiguity because it places unwarranted faith in the ability of customers to communicate project requirements or to articulate fully their business needs [3].

Innovation-intensive projects must begin by visualizing the intended outcome of a project; a changeable, high-level vision of what it must accomplish. Crisply articulated project specifications—however accurate—do not always conduce successful execution because they impose rigid, preconceived notions of how that outcome must be achieved. Instead, system-level design concepts emerge through knowledge integration, the interactive and iterative process of creatively recombining the fragmented expertise that already exists in a team [5]. This is especially uneasy in novel e-business projects in which software professionals must collaborate across domains that rarely cross their paths.

## **A Sponge for the Unexpected**

To equate meeting the needs initially identified by a customer with project success is sheer naivety because both customer needs and technology drift even in short projects [4]. A more realistic yardstick is how well a project meets its business needs at the time of delivery.

Existing software development approaches help manage well technical risk—the risk that a project does not meet its design specifications. What they do not help manage is market risk—the risk that specifications do not meet a customer’s business needs [7]. E-business projects are especially prone to market risks, which are notoriously difficult to estimate. Rapidly changing technology, fickle customers, and evolving business needs reward above all the ability of a team to manage the unexpected—to adapt and to improvise.

**Adaptation.** A project team that successfully adapts is one that satisfies the time, budget, features, and functionality constraints that exist at the time of delivery. Absorption of unexpected changes requires the ability to turn on a dime to reorient a project, constantly adjusting to new ideas, information, and unfolding events. Responsiveness to changing customer needs rather than a project manager’s specifications then determines project execution success.

**Improvisation.** The ability to devise spontaneous, “just-in-time” workarounds to unanticipated changes and failure of existing approaches is key to handling the surprises of which unpredictable markets are capable. In projects with short windows of opportunity, small delays can spell the difference between a workable solution and a perfect project that is a little too late. When a project team cannot reliably predict the unexpected, such improvisation capacitates it to think on its feet to contain the unexpected. Such impromptu workarounds emerge through coordination of expertise fragmented across business partners, vendors, and other stakeholders.

## How the Study was Conducted

Our study focused on the worst-case scenarios: 42 e-business projects in which team members did not know each other, came from radically different organizations, and were charged with building unprecedented e-business solutions in about six months on average. We chose to study only e-business projects because they are pronouncedly high-uncertainty software projects. Nevertheless, our results are as applicable to other types of software projects.

We surveyed 142 developers and managers executing these projects in a network of logistics, infrastructural technology vendors, a leading operating systems developer, several leading U.S. electronic commerce retailers, and three consulting firms. On average, our respondents had 21 months of e-business experience and eight years of IT experience. The typical project team consisted of nine members and the average project schedule was six months, with some as short as eight weeks. The questionnaire was extensively pretested with three senior IT managers, six academic domain experts, and 79 IS students. Project execution success was evaluated by managers for each project, many of whom belonged to customer firms. The analyses on team-level aggregated data were conducted using structural equation modeling. Statistically, we have over 95% confidence that our findings of the strong relationship between knowledge integration and project execution success is not by chance.

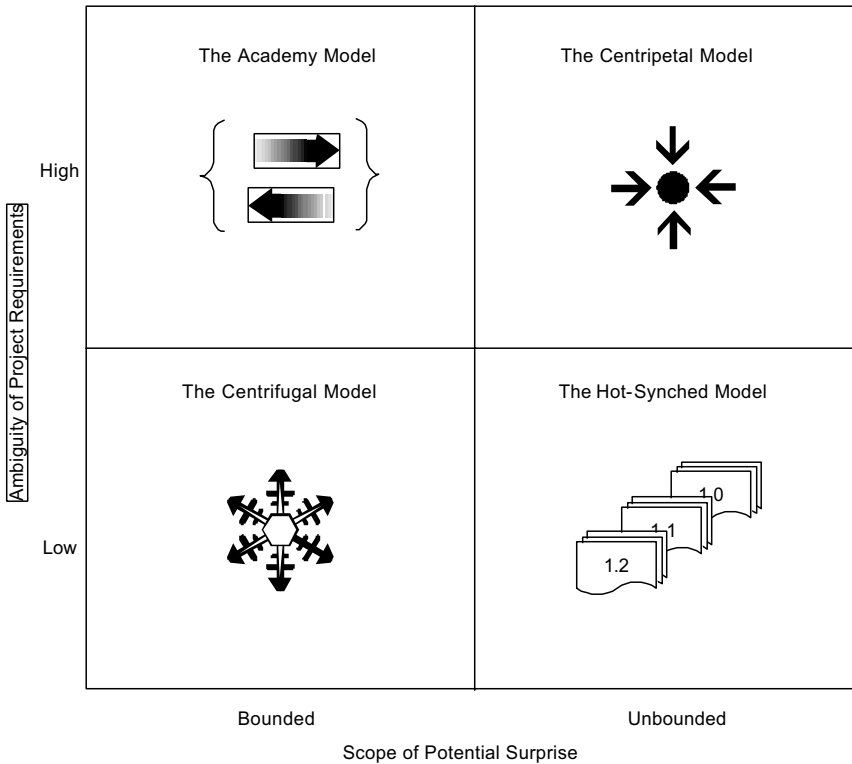
In our survey, we found a strong relationship between knowledge integration and the ability of teams to adapt and improvise during project execution.<sup>1</sup> Not only were such teams more creative, they also frequently beat deadlines. Such agility reduces market risk by ensuring that a project is responsive to its customers' evolving needs, and is successfully completed before changing markets mandate extensive rework. On the other hand, teams that slavishly follow the original plan are so preoccupied with completing a project that they fail to notice that the original target has moved. This is a seductive but dangerous trap that can, at best, deliver a good solution to a different problem.

## Managing the Unexpected

How can managers prepare their teams to manage the unexpected? Depending on the level of requirements ambiguity and the scope of potential surprise, managers can choose among four ways to organize projects (Figure 1). This choice determines whether a team is coupled loosely enough to be adaptive but tightly enough to be effective.

**The centrifugal model.** This classic model of organizing projects works best when the project requirements can be accurately identified and the scope of potential surprises is well bounded. The relatively independent activities of each team member

<sup>1</sup>In our analyses, this was supported by a statistically significant path coefficient of 0.295, which accounted for over 41% of the variance in adaptation and improvisation.



**Figure 1.** Four organizing approaches to manage the unexpected.

are guided by a central project plan that ensures that the completed pieces neatly fit together. The success of such projects hinges on rigorously following the initial project plan. Knowledge integration enters such projects only during the initial stages when their architectural features are being stabilized. Projects that incrementally extend existing systems and porting of applications fall into this category. Although this approach has served projects well in the past, it is too inflexible in dynamic project contexts.

**The academy model.** The academy model of organizing projects brings together specialists from a small number of domains. When the scope of potential surprises is limited, integration of expertise across a narrow cluster of contributing domains helps resolve ambiguous project requirements. Prototypes provide the common language through which such teams can successfully integrate knowledge from disparate domains. Such iterative prototypes generate controlled failure that further generates new information that brings into clearer view the actual business needs and constraints. Innovative, new knowledge emerges from the learning that occurs from attempting to build an executable knowledge structure which is only incompletely articulated at the outset. Examples of projects that have used this organizing logic are scientific software projects at The SAS Institute and electronic tax filing systems at H&R Block.

**The hot-synched model.** This approach is similar to the centrifugal model with the exception that the central project specifications are considered only a starting point. When project specifications can be articulated but its context is unpredictably changing, this approach ensures lock-step synchronization of team members' activities. Progressive specifications freezes ensure integration of different members' contributions while maintaining systemic interdependencies. Microsoft's sync-and-stabilize method of "daily builds" is perhaps the best recognized example of this approach. Since this model minimizes the need for explicit communication for effective knowledge integration, its success hinges on the quality of implicit coordination among team members' expertise.

**The centripetal model.** The centripetal model of organizing begins with a well-articulated vision of the intended business outcomes expected of a project. Unlike the centrifugal approach that begins with an elaborate specifications document, evolving project-level concepts are the outcome of continual knowledge integration across a diverse variety of knowledge domains. This approach best suits projects with ambiguous requirements and an unpredictable range of possible surprises. Astute orchestration of customer's and team members' diverse knowledge collaboratively creates project-level systemic knowledge that cannot be developed in a more homogenous group. This approach shuns repeatable processes, which are designed to optimize known activities by restricting unknown activities [1]. Instead, it places a premium on creativity, out-of-the-box thinking, and flexibility. In our study, we found that knowledge integration contributes most to execution success in such projects. Managers of centripetal teams must emphasize creating new ways of solving problems over acquiring new individual skills. Application of individual expertise across functional boundaries and rapid experimentation are key to its success. Although facets of a given project might fall into more than one quadrant, focus on the one that characterizes the critical path activities in the project.

### **A Manager's Checklist for Planning for the Unexpected**

- Determine the ambiguity of project requirements.
- Consider the scope of potential surprises—technology developments, changing markets, and requirements volatility.
- Match the project to the quadrant in Figure 1
- Organize the project based on one of the four approaches to manage and absorb the unexpected

### **Balancing on the Tightrope**

The key to managing the unexpected in e-business software projects is balancing flexibility and efficient execution. Our results provide some surprising insights into the value of experience, methodologies, and team size in maintaining such balance.

**General IT experience trumps e-business experience.** Novel software projects must not put excessive faith in team member's e-business experience because such context-specific experience is easily rendered irrelevant by changing technology. In dynamic environments, it generates little more than unwarranted overconfidence [6]. In contrast, we found that general IT experience is more valuable because it embodies higher-level knowledge gained through several cycles of learning across different generations of systems.<sup>2</sup>

**Methodologies are valuable.** Managers must carefully weigh abandoning time-tested software development approaches because we did not find that to be a necessary condition for project success.<sup>3</sup> Existing methodologies might be imperfect but not irrelevant to e-business projects.

**Smaller is better.** We found that smaller teams are better at knowledge integration, and consequently are more successful in executing novel software projects.<sup>4</sup> Managers should consider decomposing a larger project into smaller subprojects when feasible, assigning the bare number of necessary individuals to each team.

Managers of dynamic, innovation-intensive e-business projects must realize that bringing together the best of experts does not guarantee success. If they pragmatically attend to knowledge integration, they no longer need to fear the unexpected. Instead, they can wait for it. Prepared.

## References

1. Armour, P. Matching processes to types of teams. *Commun. ACM* 44 (2001), 21–23.
2. Armour, P. A case for a new business model: Is software a product or a medium? *Commun. ACM* 43 (2000), 19–22.
3. Armour, P. The five orders of ignorance. *Commun. ACM* 43 (2000), 17–20.
4. Benamati, J., and Lederer, A. Coping with rapid changes in IT. *Commun. ACM* 44 (2001), 83–88.
5. Grant, R. Prospering in dynamically-competitive environments: Organizational capability as knowledge integration. *Organization Science* 7 (1996), 375–387.
6. Kambil, A., and van Heck, E. Reengineering the Dutch flower auctions: A framework for analyzing exchange organizations. *Information Systems Research* 9 (1998), 1–19.
7. MacCormack, A., Verganti, R., and Iansiti, M. Developing products on internet time: The anatomy of a flexible development process. *Management Science* 47 (2001), 133–150.
8. Robillard. The role of knowledge in software development. *Commun. ACM* 42 (1999), 87–92.
9. Standish Group. *CHAOS Chronicles II*. West Yarmouth, MA, 2001.

---

<sup>2</sup>In our analyses, general IT experience had a positive and e-business experience had a negative and statistically significant effect on project execution success.

<sup>3</sup>In our analyses, the relationship between abandoning existing methodologies and project execution success was statistically non-significant.

<sup>4</sup>This was supported by a statistical negative relationship between team size and project execution success in our analyses.