



The J2EETM Architect's Handbook

*How to be a successful technical architect
for J2EETM applications*

Derek C. Ashmore



1

Project Development Team and Project Life Cycle

This chapter lays the foundation for building a successful first project, from inception to release. It begins by defining what a technical architect is and does and summarizes how the architect works with other team members. The chapter continues with a look at a few alternative approaches to the development process. Still a subject of considerable debate, the definitive process for building a successful project does not yet exist, leading many companies to adopt a hybrid plan.

Project Development Team: Roles and Responsibilities

All J2EE development teams need people with a wide variety of skills to fill numerous roles within the team. Among the many skill sets needed to make a J2EE project successful are:

- ▲ Technical architect
- ▲ Project manager
- ▲ Business analyst
- ▲ Layout designer
- ▲ Presentation-tier developer

- ▲ Business logic developer
- ▲ Data modeler
- ▲ Database administrator
- ▲ Data migration specialist
- ▲ Infrastructure specialist
- ▲ Testing specialist

Although the book focuses on the role of the technical architect, this section defines the roles and responsibilities of other major players on the J2EE development team and describes the responsibilities of the technical architect with respect to those roles.

Some organizations use different labels for the roles. For instance, an infrastructure specialist may be called a system administrator, a testing specialist may be a tester, and some organizations may distinguish between a test team manager and individual testers. Regardless of the terms you attach to these skill sets, making all of them part of a development team greatly increases its chances of creating a successful J2EE project.

Further, it's possible for one person on the team to fill many roles and for one role to be co-owned by multiple people if the project is large enough. Some organizations combine the roles of technical architect and project manager. Some organizations have a senior developer double as a database administrator or as an infrastructure specialist. And some have the same developers work on the presentation tier as well as the business layer. I'm not trying to recommend a team organization but merely to communicate what skill sets are necessary, however they are organized.

Technical Architect

The technical architect identifies the technologies that will be used for the project. In many organizations, some technology choices are made at an enterprise level. For instance, many organizations make hardware and operating system choices and some software choices (e.g., the J2EE container vendor) at an enterprise level. Commonly, choosing a language, such as Java, is an enterprise-level decision.

However, most applications have technical requirements that aren't explicitly provided in enterprise-level edicts. I make a distinction between technology choices made at an enterprise level and those made for individual applications. For example, a decision to use the Java language for all server-side programming might be made at an enterprise level, but a decision about

which XML parser to use might be left to individual application architects. In many organizations, the people making enterprise-level technology choices make up a group separate from the J2EE development team.

The technical architect is commonly responsible for identifying third-party packages or utilities that will be used for a project. For example, the architect might identify a need for a template-based generation engine and choose Apache's Velocity.

The technical architect recommends the development methodologies and frameworks of the project. Typically, the architect makes these recommendations to the project manager. For example, a common recommendation is to document all analyses in use-case format and supplement with a prototype. Another common recommendation is to document the design in terms of an object model. Some organizations define the methodologies used at an enterprise level.

The technical architect provides the overall design and structure of the application. Each developer brings to a project a unique set of preconceived opinions, habits, and preferences. Synthesizing the input of this sometimes widely divergent group, the technical architect ensures that the work done by individual developers is complementary.

I liken the role of technical architect to that of an orchestra conductor. All musicians have differing opinions about how to interpret a given work. The conductor provides the interpretation that will be used and works with the musicians to implement it.

The technical architect ensures that the project is adequately defined. A project analysis must be detailed and consistent enough to form the basis for building an application. Typically, the technical architect works with the project manager and business analyst to define the project.

The technical architect ensures that the application design is adequately documented. Documenting the application design is a critical step in establishing sound communication with and among developers. The process of creating documentation forces the architect to think through issues thoroughly. And the final document enables management to add or change developers to the project without adversely encroaching on the architect's time. For developers, documentation allows them to proceed if the technical architect is absent from the project for a limited period and enables them to work through design inconsistencies on their own without consuming the

RUP's emphasis on centralized analysis and design has great value. XP assumes that developers can take a parochial view of the story they are working on and ignore anything else. This can cause some amount of rework. All developers really should have a larger focus. Because RUP concentrates analysis and design at the beginning of a project, it represents a sensible compromise between a purely iterative approach and the waterfall approach.

It is necessary to control communication with end users. XP assumes that any member of the development team should be able to talk to an end-user representative. Developers and end users usually have different perspectives and use different terminology. In practice, many developers have trouble adapting to nontechnical terminology. They simply can't translate business terminology into technical terminology, and vice versa. Some centralization of communication to the business side is necessary as a practical matter.

Further Reading

Beck, Kent. 2000. *Extreme Programming Explained*. Reading, MA: Addison-Wesley.

Brooks, Frederick P., Jr. 1975. *The Mythical Man-Month: Essays on Software Engineering*. Reading, MA: Addison-Wesley.

Kroll, Per, and Philippe Krutchen. 2003. *The Rational Unified Process Made Easy: A Practitioner's Guide to the RUP*. Boston: Addison-Wesley.