Chapter 2.8
WebSphere Portal 6.1:
An Agile Development Approach

Thomas Stober
IBM Germany Research and Development, Germany

Uwe Hansmann
IBM Germany Research and Development, Germany

ABSTRACT

IBM's Portal technology continues to evolve as a powerful infrastructure for integrating the IT landscape, by presenting it as a consolidated view to the user community. The new capabilities of WebSphere Portal 6.1 are the outcome of a world-wide development team, which focused on this release for the past 2 years. During that time major architectural enhancements have been introduced and a significant amount of code was written. In this article the authors will describe how developers and testers have adopted agile principles to collaborate across the globe. In detail, aspects like an iterative approach, test driven development, budget based prioritization and cross-organization teaming will be discussed. The authors will also cover how "tiger teams" interact with customers by making early code drops available and responding to feedback.

REACHING THE LIMITS OF THE CLASSICAL WATERFALL APPROACH

It ain't what you don't know that gets you into trouble. It's what you know for sure that just ain't so. Mark Twain

Developing market-leading Enterprise Portal products, like WebSphere Portal requires a first-class development team. Far more than 300 developers and test engineers are working for different organizational units and collaborate in very far apart time zones. There are 8 major development sites across the world. The product has dependencies on other IBM products, such as WebSphere Application Server, and is the base for other products, like Lotus Quickr. Further dependencies arise from customer requirements and commitments.

IBM WebSphere Portal leads this wave of innovation, combining the latest user-centric functionality with reliable security and manageability features to meet the needs of the business. The software incorporates extensive Web 2.0 capabilities, allowing companies to fuel social interaction by delivering high-performing, intuitive applications through a rich Web interface. This new release adopts the latest industry-driven standards. It also introduces flexible ways to create and manage Portal sites and content. Many more enhancements emphasize increased utility and flexibility, such as web site management, integration of non-Portal pages as well as step up authentication.

Up to Portal 6.0, the Portal team used to work a classical waterfall approach. Product management captured the requirements and work items for a particular release project, and prioritized them. The project management team assembled a complex project plan with a break down of distinct task assignments for individual developers and testers. There were milestones, test phases, and fixed target dates to achieve the well defined goals. In general there was an analysis and design phase in which content, architecture, and project plan were established. In addition, there was distinct development phase executing the plan, followed by the distinct test phase. In major releases the project plan covered a period of up to 1.5 years.

However the complexity of the technology and especially the growing complexity of the team and time constraints have made it more and more difficult to execute the established plan as scheduled. Future needs and issues are difficult to predict. Each of these distinct phases turned out to be not that distinct and isolated. Instead there were dependencies, loops circulating back to earlier phases in order to adjust. Communication and interfaces between different organizational units are a challenge in large distributed teams. It is extremely difficult to make sure that the right information is made available to the right set of people. Bringing the independently developed pieces together in order to assemble a complex use case requires a significant integration effort, before the overall system reaches a satisfying level of stability. Development and testing were done by separate organizations. While the developers owned the responsibility for design, coding and unit testing, the test organization covered functional and system verification testing.

Typically, unforeseen issues, like a design flaw, or a growing number of bugs beyond the expected, or redirection of resources to other activities, made it necessary to rework the plan. Typically the problems are getting really pressing at the end in the final test phase. At that time, content removal isn’t really an option, as the code is already done, although not stable. Delaying the shipment is not a good option, as customers do rely on the promised delivery dates. And sacrificing the quality is not acceptable either. And obviously the costs of fixing problems increase significantly the later the issue is detected. It took a tremendous, costly team effort to solve the situation and ensure that a solid product is still being shipped on time.

As a result of these experiences, the limits of such a pre-planned waterfall approach became obvious: the classical approach it is too inflexible to react quickly enough to the highly dynamic constraints of a complex product within a large organization (Figure 1).

MOVING TO AGILE SOFTWARE DEVELOPMENT

One of the key reasons, why the Portal team has moved away from a classical waterfall approach used for Portal 6.0, is to gain more flexibility and improve the ability to react to changing constraints. Within a release project, the content needs to be decided as late as possible, while tested and usable pieces of functionality are to be made available as early as feasible.

Another goal is to optimize the flow of human interaction. Intensive collaboration should be