OOPSLA workshop #22 - Deploying Lightweight Processes

Karin Kolbe, Principal Consultant
Spirus Pty Ltd, Sydney, Australia
kkolbe @spirus.com.au

Much of my experience is with organisations that pay lip-service to a heavy process, but are actually anarchic and rely solely on the heroic efforts of individuals. Generally the managers have been schooled in the waterfall model and cling to the concept of heavy processes. Implementing a light-weight process to support OO development is thus a subtle balancing act between the needs and expectations of the developers, managers and users.

While I agree with the statement "keep processes as lightweight as possible and [begin] programming as early as possible", there are some points where this can be almost dangerous. The effectiveness of a process can only be evaluated by considering it in the context of the whole system development life-cycle (SDLC). For example, if the requirements gathering process is inadequate, then it does not matter how fast or well you build the wrong system. My approach is to lay solid foundations and then to relax. My list of essentials is below. They, and later the bulk of the development, are developed in a spiral fashion.

Project Vision – a short one page description is crucial reading for everyone joining the team – this saves hours of talking.

Requirements – Firstly, know what the market for the product is, and who the users are. Capture functional requirements with *essential* use cases. Essential use cases allow us to capture the essence of the requirements without getting bogged down in implementation details. If the use cases do not 'gel' into a clear picture that everyone is happy with, then stop the project. Consider the non-functional requirements.

Think &design for usability – This cannot be evolved into an application. Understanding the needs of individual roles (actors) is crucial. A clean user interface will generally re-use many components and ideas, which ultimately means less coding and testing.

Overall technical architecture – foundations must be strong. This is best expressed with text and a few key diagrams. Build a proof of technology early.

Design – use *robustness* diagrams before attempting sequence diagrams and avoid CASE tools. The act of drawing is wonderful but endless mouse-clicks are not – use pencil & paper and then either a scanner or train an admin assistant in using Visio. Draw state diagrams for the key objects.

Testing - test early, test often. Gather a test team to plan and execute the tests. Mandate test harnesses.

Supporting it all: Conduct reviews of all documents, artifacts and *every* line of code. With this in place, people work faster and the final quality is better. Store *everything* in a version-controlled central repository. Have precise, clear standards and maintain a library of examples.

The above do not give total protection against the vagaries of politics, personalities and all the problems that will arise, but they go a long way to assuring success.

Deploying a process

Regardless of whether a process is extreme or not, the single most important technique for deployment is to carefully articulate the process to *all* involved parties.