Changing Ownership

Position paper for the workshop on Human Issues of Agile Processes, OOPLSA 2001

Klaus Marquardt
Käthe-Kollwitz-Weg 14, D-23558 Lübeck, Germany
Email: marquardt@acm.org or agile@kmarquardt.de

Copyright © by Klaus Marquardt

Personal experience lets me focus on two important techniques of agile processes, namely XP: CONSTANT INTEGRATION and COMMON OWNERSHIP. Introducing these principles can reduce project risk, increase the quality and accelerate the development dramatically, but changes the way each individual engineer works and has a large impact on human issues. When the Agile Manifesto values "Individuals and interactions over processes and tools", these two techniques come with a strong aftertaste.

CONSTANT INTEGRATION removes one of the largest risks a project can face - that the system simply does not work, and that you do not even know about it. Constant Integration requires that you build and test the entire system on a fast and steady pace. It helps you to detect errors early, to create trust in your own system, and to perceive the project as "enhancing a running system" rather than "building from scratch".

But CONSTANT INTEGRATION takes something from the developer: the freedom to decide when to deliver and integrate her components to the main version, when to compile and test, and when to react on changes that peer developers induce. This loss can only be compromised by a gain in another area.

A compromise becomes possible when the turn-around times are low (less than one hour) and the environment comes with strong merge facilities. Given that, each developer can decide on her own when to deliver and when to incorporate changes. The suffering then is on her side. Her functionality does not appear in the recently integrated version, and she has to stand all the merge conflicts that arise when her peers delivered their changes in advance. This compromise will only be accepted when the suffering gets smaller quickly as soon as the way of working is only slightly adapted to CONSTANT INTEGRATION.

COMMON OWNERSHIP is a prerequisite for CONSTANT INTEGRATION, and related to other techniques like Refactoring (omitted due to brevity). Without releasing the ownership, integration problems cannot be solved.

Furthermore, COMMON OWNERSHIP allows a project to assign tasks to a developer or pair that can be implemented in isolation from the integrated version. Such a task can often not be completed without touching artifacts whose "owner" is just not available for sudden and/or frequent interruptions. The only alternative would be to corrupt the system design. So COMMON OWNERSHIP allows combining traceable project progress with quality design.

But COMMON OWNERSHIP takes something from the developer: the freedom to design and implement a component according to her own style and taste. The feeling of ownership is often an important motivational factor and a kind of mental home. Besides, it also serves as a guard against unreasonable or low-quality changes. Loosing these benefits must be compensated by even larger benefits, which probably cannot be provided by typical agile processes.

A compromise can sometimes be found when highly visible components can be distinguished from components at the systems' edge, and a different ownership policy applies. This requires an accepted system architecture and an architect, and that the dependencies between the system components are known and explicitly managed.