design

Editor: Martin Fowler ■ ThoughtWorks ■ fowler@acm.org

The

Continuous design is also known as

common functionality. Combined with our inexperience with ASP.NET, this led to a lot of duplicationeri the presentationelayer.

We onlyeriternationalized output, not input, but it still took one pair about a week and a half. The culprit was duplication. Rather than change one method that handled all input and output, we had to find all places that output was generated and change that. As with the security retrofit, this was tedious, but not particularlyehard.

[illegible]?

On all these projects, the difficulty of making changes directly related to specific design qualities. The most obvious is duplication: when a change we wanted was localized in a single class, it was trivial. When we had to modify similar code over and over, the change was tedious and took a lot longer.

Other design qualities also affected our ability to make changes. Simplicity was important. With simpler designs, we were less likely to encouiter existing code. When adding features, we were better off when there was no preexisting design to handle that feature. Adding code that doesn't exist is easy; fixing someone's preconceptions about a feature first is more costly. The sidebar lists a number of other design qualities that have made our projects easier to maritain and change.

B[illegible]

My experiments with continuous design have been very successful. I recommend that you try it oneyour projects. Before you begin, though, look at your current process. Software processes oriented arouid up-front design might not be friendly to continuous design. At a minimum, you'll need automated tests, a team-based approach to changes (such as collective code ownership), and commitment to continuously evaluating and improving your design in the face of schedule pressure.

Y