Design for Change

Design for change refers to employing design structures, techniques, practices, idioms, and patterns that allow a software systems behavior to be more easily modified in the future.

Main Benefits	Increased ability to respond and incorporate changes into the system. Reduced maintenance costs.
Keys to Success	Identifying areas that are likely to change, using information hiding, and developing a change plan.
When to Use	Designing for change is a risk-reduction practice and can be used whenever system is likely to change or evolve during the project or in its lifetime.
Main Risks	Creating an unnecessary amount of support for change, increasing short-term costs to build the software. The amount of flexibility in the design should match the areas of expected volatility of the system.

Overview

"Designing for change" is an broad label that encompasses several change-oriented design practices. These practices need to be employed early in the software lifecycle to be effective. The success of designing for change depends on identifying likely changes, developing a change plan, and hiding design decisions so that changes do not ripple through a program. Some of the change-oriented design practices are more difficult than people think, but when they are done well they lay the groundwork for long-lived programs and for flexibility that can help to minimize the schedule impacts of late-breaking change requests.

Interactions with other Best Practices

The flexibility provided by designing for change is an important part of the support needed for incremental-development practices such as evolutionary delivery and evolutionary prototyping. The change-oriented design practices also provide moderate support for reuse.

Further Reading

Parnas, David L. "On the Criteria to Be Used in Decomposing Systems into Modules," *Communications of the ACM*, v. 5, no. 12, December 1972.

Parnas, David L. "Designing Software for Ease of Extension and Contraction," *IEEE Transactions on Software Engineering*, v. SE-5, March 1979..

Parnas, David Lorge, Paul C. Clements, and David M. Weiss. "The Modular Structure of Complex Systems," *IEEE Transactions on Software Engineering*, March 1985.

McConnell, Steve. *Code Complete*. Redmond, WA: Microsoft Press, 1993.

McConnell, Steve. *Rapid Development*. Redmond, WA: Microsoft Press. 1996.