/ AntiPatterns (/antipatterns)
/ Software Architecture AntiPatterns (/antipatterns/software-architecture-antipatterns)

Swiss Army Knife

AntiPattern Problem

A Swiss Army Knife, also known as Kitchen Sink, is an excessively complex class interface. The designer attempts to provide for all possible uses of the class. In the attempt, he or she adds a large number of interface signatures in a futile attempt to meet all possible needs.

Real-world examples of Swiss Army Knife include from dozens to thousands of method signatures for a single class. The designer may not have a clear abstraction or purpose for the class, which is represented by the lack of focus in the interface.

Swiss Army Knives are prevalent in commercial software interfaces, where vendors are attempting to make their products applicable to all possible applications.

This AntiPattern is problematic because it ignores the force of managing complexity, that is, the complicated interface is difficult for other programmers to understand, and obscures how the class is intended to be used, even in simple cases. Other consequences of complexity include the difficulties of debugging, documentation, and maintenance.

Refactored Solution

Often, complex interfaces and standards are encountered that must be utilized in a software development project; therefore, it is important to define conventions for using these technologies so that management of the application's complex architecture is not compromised.

This is called creating a *profile*. A profile is a documented convention explaining how to use a complex technology. Often, a profile is an implementation plan implementing the details of the technology. With profiles, two independent developers can use the same technology, with the likelihood of achieving interoperable software.

A profile of a software interface defines the subset of the signatures that are utilized, and should include conventions for the parameter values. In other words, the profile identifies the literal values that can be passed in each parameter.

In addition, the profile may be required to define the dynamic behavior of the applications using the interfaces. This includes descriptions and specifications of sequences of execution, method calls, and exception handling.

Variations

A Swiss Army Knife differs from the Blob AntiPattern in that there may be several Swiss Army Knives in a single design. In addition, the intent of the Swiss Army Knife is that the designer is exposing complexity in a vain attempt to address all foreseeable needs for the class. The Blob is a singleton object that monopolizes the process or data in a system.

Read next

This article is taken from our book **AntiPatterns: The Survival Guide (/antipatterns-book)**.

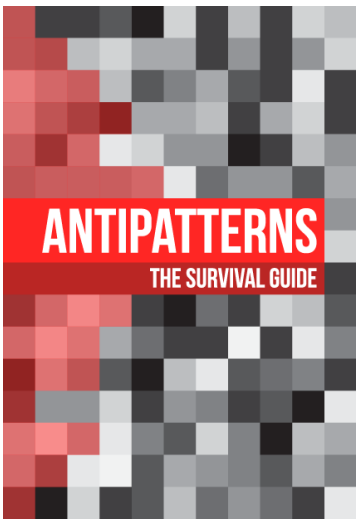
All of the AntiPatterns are compiled there. The book is written in clear, simple language that makes it easy to read and understand (just like this article).

(/antipatterns/vendor-lock-in)
- Wolf Ticket (/antipatterns/wolf-ticket)
- Architecture By Implication (/antipatterns/architecture-by-implication)
- Warm Bodies (/antipatterns/warm-bodies)
- Design By Committee (/antipatterns/design-by-committee)
- Swiss Army Knife (/antipatterns/swiss-army-knife)
- Reinvent The Wheel (/antipatterns/reinvent-the-wheel)
- The Grand Old Duke of York (/antipatterns/the-grand-old-duke-of-york)

§ Project Management AntiPatterns (/antipatterns/software-project-management-antipatterns)

- Project Management AntiPatterns (/antipatterns/software-project-management-antipatterns)
- Blowhard Jamboree (/antipatterns

We distribute it in PDF & EPUB formats so you can get it onto your iPad, Kindle, or other portable device immediately after your purchase.



(/antipatterns-book)

❤ Learn more (/antipatterns-book)

‹ Design By Committee (/antipatterns/design-by-committee)

▲ Software Architecture AntiPatterns (/antipatterns/software-architecture-antipatterns)

Reinvent The Wheel (/antipatterns/reinvent-the-wheel) ›

AntiPatterns Book (/antipatterns-book)

Design Patterns (/design_patterns)
AntiPatterns (/antipatterns) Refactoring (/refactoring)
My account (/user)
Contacts (/contact)

UML (/uml)

About us (/about-us)