Musterlösungen durch Muster-Lösungen?

„Menschliche Wesen, die die fast einzigartige Fähigkeit besitzen, aus der Erfahrung anderer zu lernen, sind gleichzeitig für ihre offensichtliche Abneigung bekannt, das auch wirklich zu praktizieren." – Douglas Adams

Vor 22 Jahren, im Spätherbst des Jahres 1994, fand zum neunten Mal die OOPSLA (Object-Oriented Programming, Systems, Languages & Applications) statt. Veranstaltungsort war die US-Metropole Portland im Bundesstaat Oregon, an der US-amerikanischen Westküste.

Die Konferenz blieb mir wegen eines geschichtsträchtigen Ereignisses in Erinnerung. Vier Pioniere hielten in einem berstend vollen Saal ein Tutorium, das den Inhalt ihres Buches zum Gegenstand hatte: *Design Patterns, Elements of Reusable Object-Oriented Software*. Bei den auch als GoF (Gang-of-Four) bekannten Autoren handelte es sich natürlich um Erich Gamma, Richard Helm, Ralph Johnson und John Vlissides. Tags drauf begann in der Ausstellungshalle der Verkauf des Buches. Derart lange Schlangen habe ich noch nie vor einem Bücherstand erlebt. Dass Entwurfsmuster während der gesamten Konferenz und auch danach zum Schwerpunkt der Gespräche avancierten, konnte daher nicht weiter überraschen.

Im Jahr darauf veröffentlichten wir auf der OOPSLA 1995 im texanischen Austin das erste Patterns-Buch der POSA-Serie (Pattern-Oriented Software Architecture). Dieses integrierte ein System aus Entwurfsmustern, Architekturmustern und Idiomen und erhielt wenig später den JOLT Productivity Award. Auch die POSA-Serie wurde zum durchschlagenden Erfolg.

In den nächsten Jahren wuchsen die Bäume quasi in den Himmel. Für die Architekten-Community galt der Siegeszug von Mustern schon als ausgemachte Sache. Bald würden für alle nicht-trivialen Entwurfsprobleme entsprechende Musterlösungen existieren, die der Informatiker nur noch aufspüren müsste. Manche Hersteller von Entwurfswerkzeugen träumten von Bibliotheken, mit deren Hilfe Entwickler und Architekten Muster auf Knopfdruck in ihren Entwurf integrieren. Mit der Möglichkeit, automatisch Muster aus Entwürfen und Softwareartefakten zu extrahieren, beschäftigten sich Vertreter der akademischen Welt.

Damals, Ende der Neunzigerjahre bis zum Anfang des 21. Jahrhunderts, entstanden viele Mythen, Fabeln und Sagen über Patterns. Eine kleine Auswahl gefällig:

- Ein Zuschauer berichtete den Teilnehmern einer Paneldiskussion zum Thema Wiederverwendung voller Stolz, er habe alle (!) Muster der „GoF-Bibel" in einer einzigen Anwendung eingesetzt. Dass er dafür Kritik erntete, hat ihn sichtlich überrascht.
- Im Code mancher Entwickler fanden sich garantiert immer wieder dieselben Lieblings-Muster, egal ob passend oder nicht. Die Moral von der Geschicht': Hammer und Nail empfiehlt sich auch bei Mustern nicht.
- Bei einem Spaziergang erläuterte mir ein Kollege, dass Muster unnötig seien, da sie sich mit ein wenig Erfahrung und Anstrengung leicht herleiten ließen. Dieser Kollege hat seine These in der Folgezeit unfreiwillig, aber durchaus eindrucksvoll widerlegt.
- Weil im Titel *„Design Patterns, Elements of Reusable Object-Oriented Software"* das Adjektiv „object-oriented" auftritt, schlussfolgerten einige Zeitgenossen, Muster seien ausschließlich für Objektorientierung anwendbar. Andere betrachteten die Muster des GoF-Buches bloß als 23 Möglichkeiten, C++-Zeiger zu verwalten.
- Manche Entwickler setzten die in den Musterbeschreibungen enthaltenen Implementierungsbeispiele einfach mit dem Muster gleich. So assoziiert das Beispiel zum State-Pattern Zustände mit Objekten. In der Praxis ist es aber unter Umständen sinnvoller, Zustände als Methoden oder Flags zu realisieren.

Nach dem Gipfel der überzogenen Erwartungen, wie ihn der Gartner Hype Cycle definiert, sollte für Muster daher schon bald der Absturz in das Tal der Enttäuschungen folgen. Und weil der Hype Cycle ein Happy End besitzt, konnten sie im Laufe der Jahre über den Pfad der Erleuchtung wieder aufsteigen; bis zum Plateau der Produktivität. Gründe gibt es dafür genug, nicht nur die Wiederverwendung von gutem Design:

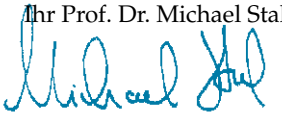
- Entwurfsmuster besitzen eine kanonische Form der Dokumentation, die sich hervorragend als mentales Werkzeug eignet, um an beliebige Entwurfsprobleme heranzugehen, beziehungsweise um deren Lösung zu beschreiben.
- Für den Umgang mit Qualitätsattributen bietet es sich an, entsprechende Entwurfsmuster und Entwurfstaktiken zu Designtaktikdiagrammen zusammenfassen, mit denen sich gezielt und systematisch Qualitätsattributszenarien umsetzen lassen.
- Da viele Muster Variabilität adressieren (Beispiele: Strategy, Bridge, Mediator, Observer, Pipes & Filters, MVC, Plug-ins), bilden sie die Grundbausteine wiederverwendbarer Software wie Bibliotheken, Produktlinien oder Ökosystemen.
- Muster helfen nicht nur beim Entwerfen von Software, sondern umgekehrt auch beim Verstehen von Entwürfen. Das gilt insbesondere deshalb, da heutige Architekturen eine relativ hohe Patterndichte aufweisen.
- Inzwischen beschränken sich Muster nicht mehr auf Blaupausen für den Softwareentwurf, sondern adressieren auch andere Aktivitäten, etwa Testen, Refaktorisierung oder Integration.

Heute gehören Muster zu den Standardwerkzeugen von Softwareentwicklern. Sie erhöhen die Produktivität, ohne als Brooksche Silberkugel zu fungieren. Sie erlauben Wiederverwendung von Expertenwissen, ohne sich auf Softwareentwurf zu beschränken. Sie bieten einen nachweisbaren Mehrwert, obwohl jedes Pattern zwar dem Entwurf dient, aber im Gegensatz zur ursprünglichen Euphorie nicht jeder Entwurf auf Mustern basiert.

Aus der Retrospektive betrachtet, gilt also auch für Patterns: Erstens kommt es anders, und zweitens als man denkt.

Ich wünsche Ihnen viel Spaß bei der Lektüre der vorliegenden Ausgabe

Ihr Prof. Dr. Michael Stal



„Experten lösen Probleme; Genies vermeiden sie." – Albert Einstein