HANSER

Leseprobe

zu

„Effektive Softwarearchitekturen“

von Gernot Starke

Print-ISBN: 978-3-446-46376-9
E-Book-ISBN: 978-3-446-46589-3
E-Pub-ISBN: 978-3-446-46690-6

Weitere Informationen und Bestellungen unter
http://www.hanser-fachbuch.de/978-3-446-46376-9
sowie im Buchhandel

Inhalt

Vorwort

Haben Sie jemals einen dummen Fehler zweimal begangen?
– Willkommen in der realen Welt.
Haben Sie diesen Fehler hundertmal hintereinander gemacht?
– Willkommen in der Software-Entwicklung.

Tom DeMarco,
in: „Warum ist Software so teuer?“

Wenn Sie sich für Baukunst interessieren, dann erkennen Sie sicherlich die „Handschrift“ berühmter Architekten wie Frank Lloyd Wright, Le Corbusier oder Mies van der Rohe immer wieder, egal, wo auf der Welt Sie auf Bauwerke dieser Meister stoßen. Die Funktionalität des Guggenheim Museums in New York oder des Opernhauses in Sydney gepaart mit deren Schönheit und Ästhetik sind unvergessliche Eindrücke. Das erwarten wir heute auch von unseren IT-Systemen: Funktionalität gepaart mit Stil!

Seit mehr als zwanzig Jahren versuche ich, Systementwicklern die Kunst des Architekturentwurfs nahezubringen. Die Erfahrung hat mich gelehrt, dass man jede Person, die mit gesundem Menschenverstand ausgestattet ist, zu einem guten Systemanalytiker ausbilden kann. Softwarearchitekten auszubilden, ist wesentlich schwieriger.

Früher waren viele unserer Systeme so einfach, dass einzelne Personen die Struktur leicht im Kopf behalten konnten. Heutzutage gehört mehr dazu, um die Struktur eines Systems zu beherrschen, die Auswirkungen von Technologieentscheidungen vorauszusehen und die Vielzahl von Hilfsmitteln wie Generatoren, Frameworks, Libraries und Entwicklungswerkzeuge kosteneffizient und zielführend einzusetzen.

Viele Jahre war ich davon überzeugt, dass nur Erfahrung in der Erstellung großer Systeme und selbst gemachte Fehler gute Architekten hervorbringen. Wir wussten einfach zu wenig über Wirkungen und Folgewirkungen von Designentscheidungen. In den letzten Jahren ist die Entwicklung von Architekturen mehr und mehr zur Ingenieursdisziplin herangereift.

Gernot Starke ist es gelungen, die Essenz dieser Disziplin auf den Punkt zu bringen. Die Tipps und Tricks, die er in diesem Buch zusammengetragen hat, vermitteln Ihnen eine Fülle von Praxiserfahrungen. Selbst wenn Sie zu den Veteranen der Branche gehören, werden Sie neben vielen Déjà-vu-Erlebnissen sicherlich noch die eine oder andere Perle entdecken. Wenn Sie gerade Ihre ersten Sporen als Architekt(in) verdienen, dann können Sie sich mit den Empfehlungen den einen oder anderen Holzweg ersparen.

Trotz aller Fortschritte in der IT bleiben Konstruktion und Ausgestaltung von Architekturen dauerhaft eine Domäne für kreative Gestaltungsarbeit von Menschen und Teams. Softwarearchitekt ist daher ein Beruf mit sicherer Zukunft!

Aachen, im September 2013

Peter Hruschka

Vorwort zur neunten Auflage

Seit der ersten Auflage 2002 sind 18 Jahre vergangen – in denen „Softwarearchitektur“ aus den Kinderschuhen zu einer volljährigen Disziplin der Softwareentwicklung reifen konnte.

Unsere Branche hat viel gelernt, Agilität und iterativ-inkrementelle Entwicklung sind endlich im Mainstream angekommen, schnelles Feedback eher Normalität denn Ausnahme, beispielsweise durch automatisiertes Testen.

Architekturaufgaben und technische Entscheidungen liegen oftmals beim Entwicklungsteam oder werden zumindest von mehreren Personen nach Rücksprache mit den Teams getroffen – und nicht mehr von Einzelpersonen „diktiert“.

Dafür sind allerdings die Anforderungen an unsere Systeme signifikant gestiegen: Benutzerzahlen von Online-Systemen gehen oft in die Millionen, Datendurchsatz oder -volumen messen wir in Tera oder Giga, nicht mehr in Kilo ... Benutzungsschnittstellen müssen hochgradig ergonomisch sein und auf multiplen Endgeräten laufen – *zero-downtime* gehört (fast) zur Selbstverständlichkeit. Embedded- und Informationssysteme müssen oftmals sicherheitskritische Entscheidungen treffen. Ach ja – Entwicklungskosten für solche komplexen IT-Systeme sollen tendenziell sinken und neue Features am besten in kürzester Zeit zur Verfügung stehen. Für solche (krassen!) Herausforderungen benötigen Entwicklungsteams fundierte Architekturkompetenz – und genau dafür habe ich dieses Buch geschrieben.

Zur neunten Auflage: Seit der achten Auflage (2017) haben viele gründliche Leserinnen und Leser einige Dutzend kleine Fehler gefunden, die in der vorliegenden Auflage (hoffentlich) behoben sind. Ich danke allen, die mich netterweise auf diese Dinge hingewiesen haben.

Die Aktualisierungen des iSAQB-CPSA-F-Lehrplans sind komplett eingeflossen, und das Kapitel über Microservices ist von den Altlasten der SOA befreit ☺.

Meinen Kollegen Dr. Lars Hupel konnte ich dazu gewinnen, einen kleinen (aber spannenden) Exkurs zum Thema „Blockchain“ beizusteuern.

Zum Buch gibt es eine (*responsive design*) Website, die ich mittels Github[1], Jekyll und Docker pflege (*http://esabuch.de*).

Möge dieses Buch helfen, bessere Software zu entwickeln!

Köln, Juni 2020

Gernot Starke

[1] Für Insider oder falls Sie Verbesserungsvorschläge haben: *https://github.com/gernotstarke/esabuch.de-site*

1 Einleitung

Wir bauen Software wie Kathedralen:
Zuerst bauen wir – dann beten wir.

Gerhard Chroust

Bitte erlauben Sie mir, Sie mit einer etwas bösartigen kleinen Geschichte zur weiteren Lektüre dieses Buchs zu motivieren.

Eine erfolgreiche Unternehmerin möchte sich ein Domizil errichten lassen. Enge Freunde raten ihr, ein Architekturbüro mit dem Entwurf zu betrauen und die Erstellung begleiten zu lassen. Nur so ließen sich die legendären Probleme beim Hausbau (ungeeignete Entwürfe, mangelnde Koordination, schlechte Ausführung, Pfusch bei Details, Kostenexplosion und Terminüberschreitung) vermeiden.

Um die für ihr Vorhaben geeigneten Architekten zu finden, beschließt sie, einigen namhaften Büros kleinere Testaufträge für Einfamilienhäuser zu erteilen. Natürlich verrät sie keinem der Kandidaten, dass diese Aufträge eigentlich Tests für das endgültige Unterfangen sind.

Nach einer entsprechenden Ausschreibung in einigen überregionalen Tageszeitungen trifft unsere Bauherrin folgende Vorauswahl:

- Wasserfall-Architektur KG, Spezialisten für Gebäude und Unterfangen aller Art,
- V&V Architektur GmbH & Co. KG, Spezialisten für Regierungs-, Prunk- und Profanbauten,
- Extremarchitekten AG.

Alle Büros erhalten identische Vorgaben: Ihre Aufgabe besteht in Entwurf und Erstellung eines Einfamilienhauses (EFH). Weil unsere Unternehmerin jedoch sehr häufig, manchmal fast sprunghaft, ihre Wünsche und Anforderungen ändert, beschließt sie, die Flexibilität der Kandidaten auch in dieser Hinsicht zu testen.

Wasserfall-Architektur KG

Die Firma residiert im 35. Stock eines noblen Bürogebäudes. Dicke Teppiche und holzvertäfelte Wände zeugen vom veritablen Wohlstand der Firmeneigner.

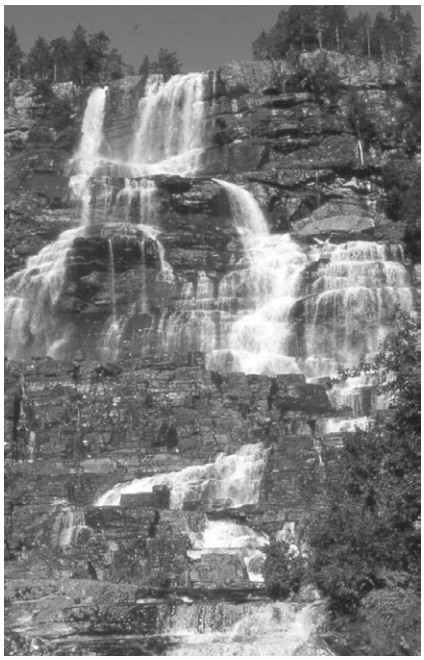


Foto von Wolfgang Korn

„Wir entwerfen auch komplexe technische Systeme“, erklärt ein graumelierter Mittfünfziger der Bauherrin bei ihrem ersten Treffen. Sein Titel „Bürovorsteher“ prädestiniert ihn wohl für den Erstkontakt zu dem vermeintlich kleinen Fisch. Von ihm und einer deutlich jüngeren Assistentin wurde sie ausgiebig nach ihren Wünschen hinsichtlich des geplanten Hauses befragt.
Als sie die Frage nach den Türgriffen des Badezimmerschranks im Obergeschoss nicht spontan beantworten kann, händigt man ihr ein Formblatt aus, das ausführlich ein Change-Management-Verfahren beschreibt.

Das Team der Wasserfall-Architektur KG legte nach wenigen Wochen einen überaus detaillierten Projektplan vor. Gantt-Charts, Work-Breakdown-Struktur, Meilensteine, alles dabei. Die nächsten Monate verbrachte das Team mit der Dokumentation der Anforderungsanalyse sowie dem Entwurf.

Pünktlich zum Ende dieser Phase erhielt die Unternehmerin einen Ordner (zweifach) mit fast 400 Seiten Beschreibung eines Hauses. Nicht ganz das von ihr Gewünschte, weil das Entwicklungsteam aus Effizienzgründen und um Zeit zu sparen einige (der Bauherrin nur wenig zusagende) Annahmen über die Größe mancher Räume und die Farbe einiger Tapeten getroffen hatte. Man habe zwar überall groben Sand als Bodenbelag geplant, könne das aber später erweitern. Mit etwas Zement und Wasser vermischt, stünden den Hausbewohnern später alle Möglichkeiten offen. Im Rahmen der hierbei erwarteten Änderungen habe das Team vorsorglich die Treppen als Rampe ohne Stufen geplant, um Arbeitern mit Schubkarren den Weg in die oberen Etagen zu erleichtern. Das Begehren unserer Unternehmerin, doch eine normale Treppe einzubauen, wurde dem Change-Management übergeben.

Die nun folgende Erstellungsphase (die Firma verwendete hierfür den Begriff „Implementierungsphase“) beendete das Team in 13 statt der geplanten acht Monate. Die fünf Monate Zeitverzug seien durch widrige Umstände hervorgerufen, wie ein Firmensprecher auf Nachfrage erklärte. In Wirklichkeit hatte ein Junior-Planning-Consultant es versäumt, einen Zufahrtsweg für Baufahrzeuge zu planen – das bereits fertiggestellte Gartenhaus musste wieder abgerissen werden, um eine passende Baustraße anlegen zu können.

Ansonsten hatte das Implementierungsteam einige kleine Schwächen des Entwurfs optimiert. So hatte das Haus statt Treppe nun einen Lastenaufzug, weil sich die ursprünglich geplante Rampe für Schubkarren als zu steil erwies. Das Change-Management verkündete stolz, man habe bereits erste Schritte zur Anpassung des Sandbodens unternommen: Im ganzen Haus seien auf den Sand Teppiche gelegt worden. Leider hatte ein Mitglied des Wartungsteams über den Teppich dann, in sklavischer Befolgung der Planungsvorgaben, Zement und Wasser

aufgebracht und mit Hilfe ausgeklügelt brachialer Methoden zu einer rotgrauen zähen Paste vermischt. Man werde sich in der Wartungsphase darum kümmern, hieß es seitens der Firma.

Die zu diesem Zeitpunkt von den Wasserfall-Architekten ausgestellte Vorabrechnung belief sich auf das Doppelte der ursprünglich angebotenen Bausumme. Diese Kostensteigerung habe die Bauherrin durch ihre verspätet artikulierten Zusatzwünsche ausschließlich selbst zu verantworten.

V&V Architektur GmbH & Co. KG

Die V&V Architektur GmbH & Co. KG (nachfolgend kurz V&V) hatte sich in den vergangenen Jahren auf Regierungs-, Prunk- und Profanbauten spezialisiert. Mit dem unternehmenseigenen Verfahren, so wird versichert, könne man garantiert jedes Projekt abwickeln. Der von V&V ernannte Projektleiter überraschte unsere Unternehmerin in den ersten Projektwochen mit langen Fragebögen – ohne jeglichen Bezug zum geplanten Haus. Man müsse unbedingt zuerst das Tailoring des Vorgehensmodells durchführen, das Modell exakt dem geplanten Projekt anpassen. Am Ende dieser Phase erhielt sie, in zweifacher Ausfertigung, mehrere Hundert Seiten Dokumentation des geplanten Vorgehens.

Dass ihr Einfamilienhaus darin nicht erwähnt wurde, sei völlig normal, unterrichtete sie der Projektleiter. Erst jetzt, in der zweiten Phase, würde das konkrete Objekt geplant, spezifiziert, realisiert, qualitätsgesichert und konfigurationsverwaltet.

Foto von Ralf Harder

Der Auftraggeberin wurde zu diesem Zeitpunkt auch das „Direktorat EDV“ der Firma V&V vorgestellt. Nein, diese Abteilung befasste sich nicht mit Datenverarbeitung – die Abkürzung stand für *„Einhaltung Des Vorgehensmodells“*.

Nach einigen Monaten Projektlaufzeit stellte unsere Bauherrin im bereits teilweise fertiggestellten Haus störende signalrote Inschriften auf sämtlichen verbauten Teilen fest. Das sei urkundenechte Spezialtinte, die sich garantiert nicht durch Farbe oder Tapete verdecken ließe, erklärte V&V stolz. Für die Qualitätssicherung und das Konfigurationsmanagement seien diese Kennzeichen unbedingt notwendig. Ästhetische Einwände, solche auffälligen Markierungen nicht in Augenhöhe auf Fenster, Türen und Wänden anzubringen, verwarf die Projektleitung mit Hinweis auf Seite 354, Aktivität PL 3.42, Paragraph 9 Absatz 2 des Vorgehensmodells, in dem Größe, Format, Schrifttyp und Layout dieser Kennzeichen verbindlich definiert seien. Die Bauherrin hätte bereits beim Tailoring widersprechen müssen, nun sei es wirklich zu spät.

Extrem-Architekten AG

Die Extrem-Architekten laden unsere Unternehmerin zu Projektbeginn zu einem Planungsspiel ein. Jeden Raum ihres geplanten EFH soll sie dabei der Wichtigkeit nach mit Gummi-

bärchen bewerten. Die immer nur paarweise auftretenden Architekten versprechen ihr eine erste funktionsfähige Version des Hauses nach nur sechs Wochen. Auf Planungsunterlagen würde man im Zuge der schnellen Entwicklung verzichten.

„Gummibär-Tango“ von Klaus Terjung

Zu Beginn der Arbeiten wurde das Team in einer Art Ritual auf die gemeinsame Vision des Hauses eingeschworen. Wie ein Mantra murmelten alle Teammitglieder ständig mit seltsam gutturaler Betonung die Silben „Einfa-Milien-Haus“, was sich nach einiger Zeit zu „Ei-Mi-Ha“ abschliff. Mehrere Außenstehende wollen gehört haben, das Team baue einen bewohnbaren Eimer. Sie stellten eine überdimensionale Tafel am Rande des Baugeländes auf. Jeder durfte darauf Verbesserungsvorschläge oder Änderungen eintragen. Dies gehöre zu einem Grundprinzip der Firma: „Kollektives geistiges Eigentum: Planung und Entwurf gehören allen.“

Nach exakt sechs Wochen laden die Extrem-Architekten die Unternehmerin zur Besichtigung der ersten funktionsfähigen Version ein. Wieder treten ihr zwei Architekten entgegen, jedoch erkennt sie nur einen davon aus dem Planungsspiel wieder. Der andere arbeitet jetzt bei den Gärtnern. Der ursprüngliche andere Gärtner hilft dem Elektriker, ein Heizungsbauer entwickelt dafür die Statik mit. Auf diese Weise verbreite sich das Projektwissen im Team, erläutern beide Architekten eifrig.

Man präsentiert ihr einen Wohnwagen. Ihren Hinweis auf fehlende Küche, Keller und Dachgeschoss nehmen die Extrem-Architekten mit großem Interesse auf (ohne ihn jedoch schriftlich zu fixieren).

Weitere sechs Wochen später hat das Team eine riesige Grube als Keller ausgehoben und den Wohnwagen auf Holzbohlen provisorisch darüber befestigt. Das Kellerfundament haben ein Zimmermann und ein Statiker gegossen. Leider blieb der Beton zu flüssig. Geeignete Tests seien aber bereits entwickelt, dieser Fehler käme garantiert nie wieder vor.

Mehrere weitere 6-Wochen-Zyklen gehen ins Land. Bevor unsere Unternehmerin das Projekt (vorzeitig) für beendet erklärt, findet sie zwar die von ihr gewünschte Küche, leider jedoch im Keller. Ein Refactoring dieses Problems sei nicht effektiv, erklärte man ihr. Dafür habe man im Dach einen Teil der Wohnwagenküche verbaut, sodass insgesamt die Zahl der Küchen-Gummibären erreicht worden sei.

Das immer noch flüssige Kellerfundament hat eines der Teams bewogen, auf die Seitenwände des Hauses auf Dauer zu verzichten, um die Lüftung des Kellers sicherzustellen. Im Übrigen besitzt das Haus nur ein Geschoss, das aktuelle Statik-Team (bestehend aus Zimmermann und Gärtner) hat dafür die Garage in drei Kinderzimmer unterteilt.

Weil das Team nach eigenen Aussagen auf die lästige und schwergewichtige Dokumentation verzichtet hatte, waren auch keine Aufzeichnungen der ursprünglichen Planung mehr erhalten.

Im Nachhinein beriefen sich alle Projektteams auf ihren Erfolg. Niemand hatte bemerkt, dass die Bauherrin keines der „implementierten“ Häuser wirklich akzeptierte.

Chaos nur am Bau?

Keineswegs! Ähnlichkeiten mit bekannten Vorgehensweisen bei der Softwareentwicklung sind ausdrücklich gewollt, denn nicht nur beim Hausbau herrscht Chaos.[1] Auch andere Ingenieurdisziplinen erleben *turbulente Situationen*, obwohl der Maschinenbau über mehr als 200 Jahre Erfahrung verfügt. In der Softwarebranche geht es mindestens ebenso schlimm zu.

Der regelmäßige Chaos-Report der Standish-Group zeigt eine seit Jahren gleichbleibende Tendenz: Über 30 % aller Softwareprojekte werden (erfolglos) vorzeitig beendet, in über 50 % aller Softwareprojekte kommt es zu drastischen Kosten- oder Terminüberschreitungen.[2]

1.1 Softwarearchitekt(inn)en

Liebe Softwarearchitektinnen – ich meine überall auch Sie – obwohl ich im weiteren Buch durchgängig die maskuline (weil kürzere) Form „Softwarearchitekt“ verwende. Ich meine grundsätzlich Softwarearchitektinnen und Softwarearchitekten – verzichte aber auf künstliche Konstrukte wie Softwarearchitekt*innen.
Danke für Ihr Verständnis!

Softwarearchitekten allein können diese Probleme nicht lösen. Stakeholder mit klaren Zielvorstellungen, ein motiviertes Entwicklungsteam und ein effektives, flexibles und (hoffentlich) agiles Management bilden wichtige Voraussetzungen für erfolgreiche Softwareentwicklung[3].

Die Rolle „Softwarearchitektur“ kommt in Softwareprojekten besondere Bedeutung zu:

Softwarearchitekten bilden die Schnittstelle zwischen Analyse, Entwurf, Implementierung, Management und Betrieb von Software.

Architekten als Anwälte der Kunden

Diese verantwortungsvolle Schlüsselrolle bleibt in vielen Projekten oft unbesetzt oder wird nicht angemessen ausgefüllt. Architekten sollten als „Anwälte der Kunden“ arbeiten. Sie sollen sicherstellen, dass die Anforderungen der Kunden einerseits umsetzbar sind und andererseits auch umgesetzt werden.

Langfristigkeit

Softwarearchitekten denken langfristig – auf die gesamte Lebensdauer von IT-Systemen bezogen. Sie ermöglichen kurzfristige Änderungen, sichern gleichzeitig die Langlebigkeit und Nachhaltigkeit von Software.

---

1 Viele Grüße nach Berlin. Ähnlichkeiten mit gescheiterten Flughafenprojekten sind rein zufällig.

2 Quelle: The Standish Group Chaos Report. Erhältlich unter *www.standishgroup.com*

3 Eine gute Voraussetzung für Projekterfolg ist es, im Team die Eigenschaften **Kompetenz**, **Energie** und **Verantwortung** zu bündeln – danke an Dierk König (@mittie) für diese Formulierung.

Konzeptionelle Integrität

Softwarearchitekten verfolgen konzeptionelle Integrität (auch genannt *Konsistenz*): Die gesamte Konstruktion von Software sollte einem einheitlichen Stil folgen. Insbesondere sollten *ähnliche Aufgabenstellungen in Systemen ähnlich gelöst* werden. Dies erleichtert das Verständnis und die langfristige Weiterentwicklung.

Das Buch gibt aktiven und angehenden Softwarearchitekten praktische Ratschläge und Hilfsmittel, diese komplexe Aufgabe effektiver zu erfüllen. Es unterbreitet konkrete Vorschläge, wie Sie als Softwarearchitekt in der Praxis vorgehen sollten. Auch wenn Sie in anderen Funktionen in Softwareprojekten arbeiten, kann dieses Buch Ihnen helfen. Sie werden verstehen, welche Bedeutung Architekturen besitzen und wo die Probleme beim Entwurf von Architekturen liegen.

1.2 Effektiv, agil und pragmatisch

Effektivität, Agilität und Pragmatismus prägen die Grundhaltung erfolgreicher Softwarearchitekten.

Agilität ist notwendig

Software wird in vielen Projekten immer noch als starres, unveränderliches Produkt betrachtet, obwohl Anwender und Auftraggeber laut nach hochgradig flexiblen Lösungen rufen. In der Praxis ähnelt die Softwareentwicklung leider oftmals eher dem Brückenbau: Eine Rheinbrücke bleibt auch in den kommenden Jahren eine Rheinbrücke. Weder verändert sich der Flusslauf noch wird aus einer Eisenbahnbrücke eine Startbahn für Passagierflugzeuge. Für Software stellt sich die Lage ganz anders dar: Hier kann aus einem abteilungsinternen Informationssystem schnell eine global genutzte Internet-E-Business-Lösung entstehen.

Langjährige Untersuchungen ergeben, dass sich 10 bis 25 % der Anforderungen an Software pro Jahr ändern (Quelle: Peter Hruschka). Management und Architekten von Softwareprojekten müssen sich durch flexible und bedarfsgerechte Vorgehensweisen darauf einstellen. Das Schlüsselwort lautet „Agilität“.

Agilität und flexibles Vorgehen wird in Softwareprojekten an vielen Stellen dringend benötigt:

- Requirements Manager müssen in Projekten flexibel mit Änderungen von Anforderungen umgehen.
- Softwarearchitekturen müssen stabile Grundgerüste bereitstellen, die die Umsetzung neuer und geänderter Anforderungen ermöglichen. In der heutigen Marktsituation müssen solche Änderungen schnell und effektiv erfolgen – oder sie sind wirkungslos.
- Projekt- und Produktmanager müssen in der Lage sein, während der Erstellung eines Systems flexibel auf neue Anforderungen, neue Technologien oder aktualisierte Produkte zu reagieren. Hier bietet agiles Vorgehen und risikobasiertes Projektmanagement viele Vorteile gegenüber den strikt am Vorgehensmodell orientierten konventionellen Methoden.
- Dokumentation muss sich an spezifischen Projektbedürfnissen orientieren statt an fix vorgegebenen Ergebnistypen. Inhalt ist in flexiblen Projekten wichtiger als Form.

- Agilität erfordert allerdings auch hohe Qualifikation und Professionalität der Beteiligten. Wenn Sie in einem agilen Projekt arbeiten, müssen Sie in allen Belangen mitdenken und Verantwortung übernehmen.

Insgesamt zählt in einem Projekt nur das Ergebnis. Selten kümmern sich Anwender oder Auftraggeber im Nachhinein um die Einhaltung starrer Vorgehensmodelle.

Softwarearchitekten müssen in ihrer Funktion als Schnittstelle zwischen den Projektbeteiligten diesen Ansatz der Agilität aufnehmen und in der Praxis umsetzen.

Handeln Sie agil!

Von Peter Hruschka

b-∀gile

Agil heißt beweglich und flexibel sein. Mitdenken statt „Dienst nach Vorschrift" und Dogma.

Kein Vorgehensmodell passt für alle Arten von Projekten. Eine agile Vorgehensweise beurteilt das jeweilige Risiko der unterschiedlichen Aufgaben bei der Softwareentwicklung und wählt dann die geeigneten Maßnahmen. Folgende Schwerpunkte werden dabei gesetzt:

- offen für Änderungen statt Festhalten an alten Plänen,
- eher ergebnisorientiert als prozessorientiert,
- „miteinander darüber reden" statt „gegeneinander schreiben",
- eher Vertrauen als Kontrolle,
- Bottom-up „Best Practices" austauschen und etablieren statt Top-down-Vorgaben diktieren.

Trotzdem heißt „Agilität" nicht, Anarchie zuzulassen:

- Agile Vorgehensmodelle haben Ergebnisse, nur unterscheiden sich diese für unterschiedliche Projekte in Anzahl, Tiefgang und Formalismus.
- Agile Entwicklung kennt Prozesse, nur lassen diese mehr Spielraum für Alternativen und Kreativität.
- Agile Methoden setzen auf Verantwortung; es werden nur notwendige Rollen besetzt.
- Agile Methoden basieren auf Feedback und Iterationen. Fordern und geben *(push und pull)* Sie Rückmeldung zu Ergebnissen – nur so können Teams und Ergebnisse besser werden.

Das Risiko entscheidet: Jeder Projektleiter, Systemanalytiker, Architekt, Tester und Entwickler überprüft ständig Risiken und entscheidet in seinem Umfeld über notwendige Maßnahmen, damit aus Risiken keine Probleme werden.

*Dr. Peter Hruschka (*hruschka@b-agile.de*) ist unabhängiger Trainer und Methodenberater. Er ist Prinzipal der Atlantic Systems Guild, eines internationalen Think Tank von Methodengurus, deren Bücher den State of the Art wesentlich mitgestaltet haben (www.systemsguild.com).*

Effektiv = Ziele erreichen

Weil das Begriffspaar „effektiv und effizient“ immer wieder für Missverständnisse sorgt, möchte ich die Bedeutung beider Wörter hier kurz gegenüberstellen.

Effizient = hoher Wirkungsgrad

Eine Lexikondefinition des Begriffs „Effizienz“ lautet „Wirkungsgrad“, also das Verhältnis von Aufwand zu Ertrag. Wenn Sie Aufgaben effizient erledigen, dann arbeiten Sie also mit hohem Wirkungsgrad. Sie investieren für den gewünschten Ertrag einen minimalen Aufwand. Spitzensportler etwa vermeiden in ihren Disziplinen überflüssige Bewegungen oder Aktionen, was in hochgradig effizientem Ausführen der jeweiligen Sportart resultiert.

Prägnant ausgedrückt, bedeutet das:

Effizient = Dinge richtig machen

Effektiv = zielorientiert

„Effektiv“ bedeutet zielorientiert. Sie arbeiten effektiv, wenn Sie Dinge erledigen, die zur Erreichung Ihrer konkreten Ziele notwendig sind. Auch für diesen Begriff wieder eine prägnante Definition:

Effektiv = die richtigen Dinge machen

Es ist viel wichtiger, die richtigen Dinge zu erledigen, als irgendwelche Dinge besonders effizient zu tun.

Softwarearchitekten müssen in hohem Maße effektiv arbeiten. Kunden und Auftraggeber bestimmen Ziele, Architekten müssen sicherstellen, dass diese Ziele auch erreicht werden.

Effektiv = agil und angemessen

Effektiv bedeutet auch, angemessen und bedarfsgerecht zu agieren. Auf die Entwicklung von Software angewandt, heißt das, sich permanent an den Bedürfnissen der Kunden und Auftraggeber zu orientieren (und nicht starr an den Buchstaben eines formalen Vorgehensmodells). Ich plädiere in diesem Buch für Agilität in diesem Sinne.

Effektive Softwarearchitektur bedeutet, das (für Kunden und Auftraggeber) richtige System zu konstruieren und zu bauen – mit technisch und organisatorisch angemessenen Mitteln.

Effektiv = pragmatisch

Projekterfolg wird grundsätzlich vom Auftraggeber beurteilt, nicht vom Architekten. Auftraggeber wollen in erster Linie ein produktives System erhalten und nicht die Einhaltung eines starren Vorgehensmodells erzwingen. Architekten müssen daher den Zweck des Systems im Auge behalten.

Pragmatisches Vorgehen bedeutet:

- auf Dogmen und starre Vorschriften zu verzichten, wo sie nicht angemessen sind,
- zielorientiert (= effektiv) Lösungen im Sinne der Kunden zu entwickeln,
- auf Perfektionismus zu verzichten. 80 % des Ertrags erreicht man mit 20 % des Aufwands.

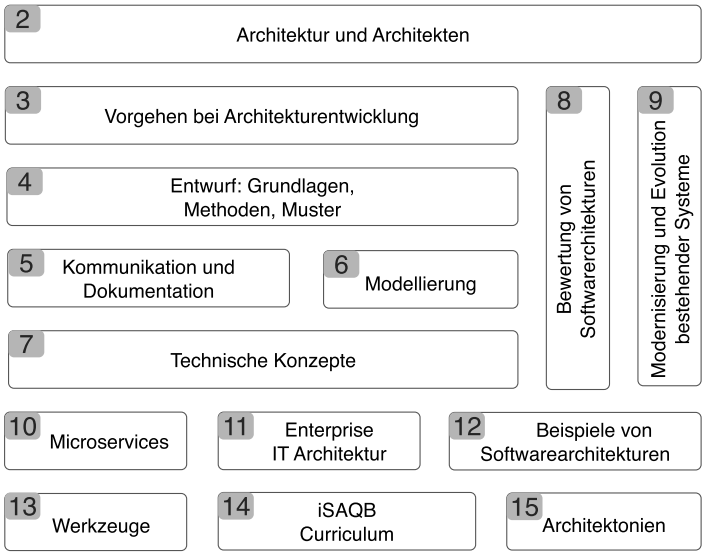
1.3 Wer sollte dieses Buch lesen?

Grundsätzlich können alle Stakeholder[4] von diesem Buch profitieren und erhalten Antworten auf zentrale Fragen.

- Softwarearchitekten
  - Was sind die Methoden und Werkzeuge unserer Zunft?
  - Wie gehen Sie beim Entwurf von Architekturen sinnvoll vor?
  - Welche praktisch erprobten Heuristiken und Ratschläge gibt es?
  - Wie meistern Sie externe Einflussfaktoren, die den Entwurf von Architekturen erschweren?
  - Welche Sichten auf Architekturen benötigen Sie in der Praxis?
- Softwareentwickler(innen)
  - Wie hängen Architektur und Implementierung zusammen?
  - Wie kann das gesamte Entwicklungsteam bei Entwurf und Pflege der Architektur konstruktiv mitwirken?
  - Welche allgemeinen Prinzipien von Softwarearchitektur und -entwurf sollten Sie bei der Implementierung unbedingt befolgen?
- Alle Personen, die sich auf die Prüfung zum „Certified Professional for Software Architecture – Foundation Level“ (CPSA-F) des iSAQB e. V. vorbereiten
- Technische Manager(innen)
  - Warum sollen Sie für Architektur Geld ausgeben? Warum ist Softwarearchitektur wichtig?
- Product-Owner, Scrum-Master, Projektleiter(innen)
  - Was genau bewirkt Architektur in Entwicklungsprojekten?
  - Welche Aufgaben erfüllen Softwarearchitekten?
  - Welche Bedeutung hat die Dokumentation von Architekturen („Was bedeuten all diese Symbole?“)?
  - Welche grundlegenden Lösungsansätze gibt es für Architekturen?
- Business-Analysten und Requirements-Engineers
  - Was bedeuten all diese Begriffe, die das Entwicklungsteam und die Architekten ständig verwenden?
  - Wie können Sie bei der Architekturentwicklung konstruktiv beitragen?

[4] Im Verlaufe des Buchs benutze ich den Begriff Stakeholder für Personen oder Organisationen, die bei der Erstellung des Systems mitwirken, es beeinflussen oder am entwickelten System ein fachliches, technisches oder kommerzielles Interesse haben.

1.4 Wegweiser durch das Buch



Wegweiser

Kapitel 2 erläutert die Begriffe „Architektur und Architekt“. Es beantwortet die Fragen nach dem Was, Warum und Wer von Softwarearchitekturen.

Kapitel 3 legt den Grundstock für eine flexible und systematische Vorgehensweise zum Entwurf von Softwarearchitekturen. Es beschreibt die wesentlichen Aktivitäten bei der Architekturentwicklung, das Vorgehen „im Großen“. Insbesondere erfahren Sie, wie Sie aus den Faktoren, die eine Architektur beeinflussen, angemessene Lösungsstrategien ableiten können.

Kapitel 4 widmet sich dem Entwurf von Systemen. Sie lernen hier die Grundlagen des Entwurfs, geltende Prinzipien und erprobte Heuristiken. Sie finden neben Methoden für die Strukturierung Ihrer Systeme und Bausteine sowie der Gestaltung von Schnittstellen auch die wichtigsten Architektur- und Entwurfsmuster, die Sie in unterschiedlichen Bereichen der Softwareentwicklung anwenden können. Dieses Kapitel gibt Ihnen die methodischen Werkzeuge für zielgerichtete Entwurfsentscheidungen an die Hand.

arc42 Template

Kapitel 5 beschreibt, wie Sie mit praxisorientierten Sichten Ihre Softwarearchitekturen kommunizieren und dokumentieren. Jede Sicht beschreibt das System aus einer anderen Perspektive. Außerdem finden Sie hier einige Hinweise für gute Architekturdokumentation sowie das bekannte Architekturtemplate arc42.

Passend dazu fasst **Kapitel 6** die Grundlagen der Modellierung für Softwarearchitekten zusammen. Sie finden u. a. einige Basiskonstrukte der UML, abgestimmt auf die Bedürfnisse von Softwarearchitekten.

(technische) Konzepte

Kapitel 7 enthält einen Katalog häufig benötigter technischer Konzepte. Hierzu zählen Persistenz (Datenspeicherung), Integration, Verteilung, Kommunikation, Sicherheit, grafische Benutzeroberflächen, übergreifende

Stichwortverzeichnis