AGILITÄT UND ARCHITEKTUR: AUS DEM ALLTAG EINES AGILEN SOFTWAREARCHITEKTEN

Wenn Ihr Chef Sie fragt, ob Sie die Rolle des agilen Softwarearchitekten übernehmen möchten, dann sagen Sie sofort zu. Denn die Aufgaben, die auf Sie warten, sind herausfordernd und spannend zugleich. Aber aufgepasst: Ein agiler Softwarearchitekt braucht neben dem Wissen über Trends und Technologien auch große Ohren, um den verschiedenen Stakeholdern gut zuhören zu können. Und gute Augen, um beim hektischen Alltag den Überblick nicht zu verlieren. Dieser Artikel erläutert die vielfältigen Aufgaben eines Architekten in einem agilen Softwareentwicklungsteam und benennt die Fähigkeiten, die für einen reibungslosen Projektablauf gefordert sind.

der autor

Urs Enzler
(urs.enzler@bbv.ch)
unterstützt neben seiner Haupttätigkeit als Softwarearchitekt bei der bbv Software Services AG Unternehmen bei der Einführung agiler Entwicklungsmethoden und -praktiken, wie z. B. Scrum und testgetriebene Entwicklung.

Die Zeiten sind vorbei, als der Softwarearchitekt in einem dunklen Raum saß und aus den Anforderungen eine Architektur zauberte. Kein Anforderungsingenieur erarbeitet die Liste der gewünschten Funktionalität im Voraus. Und die Architektur wird nicht mehr als Dokument an die Entwickler zur Umsetzung weitergegeben. In der heutigen Welt dauert der sequenzielle Ablauf dieser Phasen schlicht zu lange und ist zu träge, um am Markt bestehen zu können. Das Bedürfnis nach schnelleren Entwicklungszyklen und mehr Flexibilität bei Veränderungen der Anforderungen hat in den letzten Jahren den agilen Entwicklungsmethoden zum Siegeszug verholfen.

Doch viele Teams hadern mit der Umsetzung der Kundenwünsche. Technologische Entscheidungen gehen an den Bedürfnissen der Stakeholder vorbei, mit der Folge, dass sich die Stakeholder missverstanden fühlen – zwei Welten, die sich nicht verstehen. Hier kommt der agile Softwarearchitekt ins Spiel, dessen Aufgaben ich in diesem Artikel darstellen möchte.

Aufgabe 1: Technologische Führung

Der agile Softwarearchitekt übernimmt eine Führungsrolle für die technischen Aspekte des Projekts (siehe Abbildung 1).

Technologie-Evangelist

Der Softwarearchitekt ist der Technologie-Evangelist des Teams. Er kennt die Trends und weiß, welche Technologien in Zukunft wichtig für das Produkt und das Team sind, und er hat Argumente, um das Team und dessen Umfeld zu überzeugen, wann und warum neue Technologien eingeführt werden müssen.

Software-Engineering-Praktiken

In der agilen Welt ist es unerlässlich, dass die Architektur flexibel bleibt, um den sich verändernden Kundenanforderungen folgen zu können. Damit eine Architektur änderbar bleibt, sind gute Engineering-Praktiken gefordert. Der Softwarearchitekt übernimmt hier eine Führungsrolle und fördert den Einsatz von testgetriebener Entwicklung, akzeptanztestgetriebener Entwicklung, Pair Programming und Konzepten wie Clean Code.

Spikes

Ein Spike ist ein minimaler Prototyp, der sämtliche Schichten des Systems durchsticht. Er dient als Grundlage für Aufwandsschätzungen, Risikoabwägungen und zur Erhöhung des Wissensstands. Der Softwarearchitekt nutzt Spikes, um die architektonische Vorarbeit für das Team zu leisten, damit dieses anschließend effizient und effektiv vorwärts arbeiten kann.

Nicht-funktionale Tests

Ein Hauptaspekt der Architektur ist, dass sie es ermöglicht, die gewünschte Funktionalität mit den geforderten nicht-funktionalen Eigenschaften umzusetzen. Darum ist es die Aufgabe des Architekten, dafür zu sorgen, dass die nicht-funktionalen Anforderungen eingehalten werden. Am besten geschieht dies durch das Schreiben von automatisierten Tests. So kann jederzeit überprüft werden, ob das System die Anforderungen erfüllt.

Code schreiben

Der agile Softwarearchitekt verbringt einen großen Anteil seiner Arbeitszeit mit Programmieren. Nur so versteht er die Probleme des Teams und kann konkrete Lösungen erarbeiten. Und nicht zuletzt

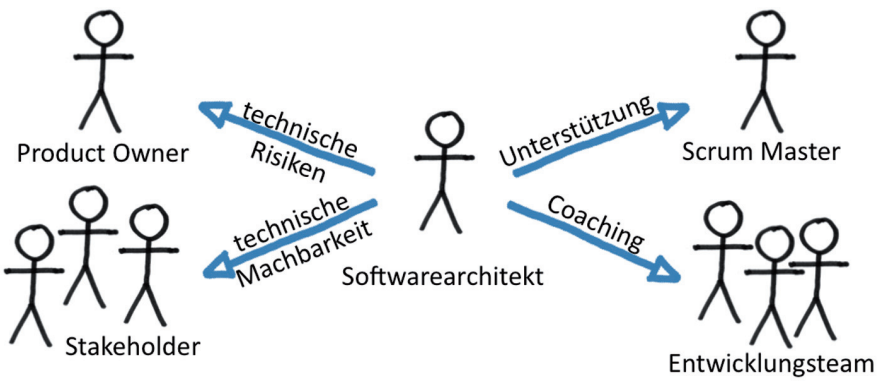

Abb. 1: Der Softwarearchitekt kommuniziert mit allen Projektbeteiligten.

Wenn mehrere Teams an einem Produkt oder Projekt arbeiten, müssen diese einer einzigen Architektur folgen. Diese Architektur muss auch in die bestehende IT-Umgebung eingebettet werden können. Die Architekten der einzelnen Teams sind daher gefordert, zusammen die Leitplanken zu definieren.

Kasten 1: Mehrere Teams und die Projektumgebung.

bestimmt der Code, wie die Architektur ist, und nicht das Diagramm an der Wand.

Aufgabe 2:
Stakeholder verstehen

Die beste technologische Lösung hilft nichts, wenn sie das falsche Problem löst. Darum engagiert sich der agile Softwarearchitekt als Bindeglied zwischen Stakeholdern und Entwicklungsteam.

Big Picture
Der Architekt ist die Person, die das Big Picture verstehen muss. Alle Aspekte – wie Funktionalität, nicht-funktionale Eigenschaften, technische Risiken und Wartbarkeit der Lösung – müssen berücksichtigt werden. Während der Entwicklung verlieren sich Entwickler schnell in Details und müssen ans Ganze erinnert werden.

Mit allen Stakeholdern sprechen
Um die Stakeholder zu verstehen, muss der Architekt mit allen sprechen:

- *Sponsor:* Was will er für sein Geld haben?
- *Endbenutzer:* Was sind Lösungen für seine Probleme?
- *Betrieb:* Was unterstützt den Betrieb des Produkts?
- *Entwickler:* Wie kann die Entwicklung vereinfacht werden?

Im Gegensatz zum *Product Owner (PO)* oder zum Anforderungsingenieur steht beim Softwarearchitekten die technische Lösung im Vordergrund und nicht die Anforderung als solche.

Alle Standpunkte kennenlernen
Der agile Softwarearchitekt spricht aber nicht nur mit den Stakeholdern, sondern er begleitet sie bei ihrer Arbeit, um besser zu verstehen, wie eine Lösung für alle angestrebt werden kann.

Den Benutzer verstehen
Besonders wichtig ist es, die Endbenutzer zu verstehen. Usability-Tests und Benutzerakzeptanz-Tests sind die Grundlage für Entscheidungen, wie die Anforderungen umgesetzt werden sollen. Der Architekt begleitet diese Arbeiten, um Auswirkungen auf die Architektur frühzeitig abschätzen zu können.

Unterstützung des Product Owners
Mit seinem technischen Wissen und dem Verständnis für die Bedürfnisse der Stakeholder unterstützt der agile Softwarearchitekt den PO bei dessen Arbeit. Er hilft dabei, Anforderungen zu erkennen, denn vor allem nicht-funktionale Anforderungen gehen gerne vergessen, und das Produkt-Backlog zu konsolidieren und zu priorisieren (Risikoüberlegungen).

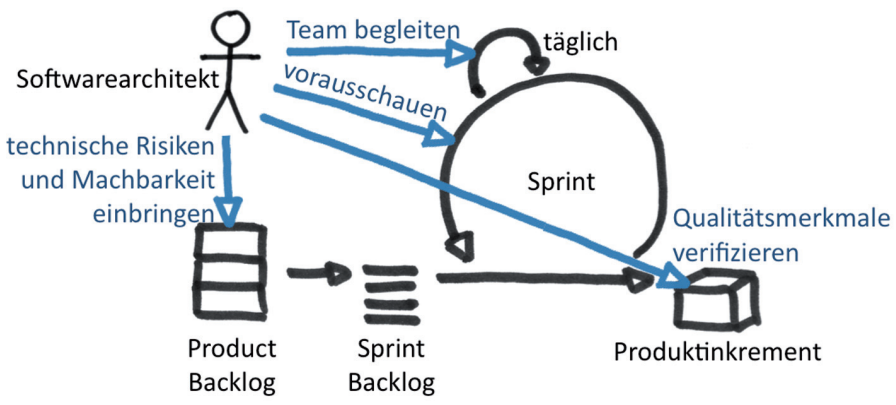

Abb. 2: Architekturtätigkeiten

Buchdicke Architekturdokumente, vollgestopft mit UML-Diagrammen, eignen sich schlecht für die Kommunikation innerhalb des Teams. Dennoch ist es unumgänglich, die Architektur zu dokumentieren – sonst werden heute gefällte Entscheide bereits bald erneut diskutiert und das Team muss die Entscheidungsgrundlage wieder erarbeiten. Hierbei helfen Templates, wie beispielsweise [Hru13].

Kasten 2: Wie viel Dokumentation genügt?

Aufgabe 3:
Teamcoach

Die Ansprüche an ein agiles Entwicklungsteam sind hoch. Der agile Softwarearchitekt gibt nicht nur sein Wissen weiter, sondern hilft dem Team auch, zusammen zu lernen.

Architektur ist Teamarbeit
Der agile Softwarearchitekt erstellt die Architektur nicht allein. Vielmehr überlässt er dies dem Team und coacht es bei der Erstellung. Bei einem agilen Vorgehen werden jeden Tag dutzende design- und architekturrelevante Entscheidungen getroffen. Alle Teammitglieder müssen in der Lage sein, korrekte Entscheidungen selbstständig zu treffen.

Koordinieren
Um die Architektur und das Design einheitlich zu halten, ist es die Aufgabe des Architekten, zwischen allen Teammitgliedern zu koordinieren (für Projekte mit mehreren Teams siehe Kasten 1).

Pair Programming
Ein gutes Mittel, um diese Koordination im Alltag umzusetzen, ist das Pair Programming. Der Softwarearchitekt kann durch häufiges Pair Programming mit allen Teammitgliedern dafür sorgen, dass einheitliche Lösungen umgesetzt werden und so ein gemeinsames Teamverständnis entsteht, wie wiederkehrende Aufgaben gelöst werden sollen.

Ausbildung des Teams
Bei kritischen Design- und Architekturentscheidungen kann der Softwarearchitekt

Ob der agile Softwarearchitekt eine Rolle ist, die durch mehrere Personen wahrgenommen wird, oder eine einzelne Person ist, hängt vom Team ab. In sehr erfahrenen Teams bringt die Verteilung der Architektenrolle auf mehrere Personen große Vorteile bezüglich Verfügbarkeit und Wissensverteilung. In Teams mit vielen unerfahrenen Mitgliedern kann die Fokussierung auf eine Einzelperson zu einfacheren Strukturen und einer klareren Linie führen – oder es ist schlicht und einfach nur eine Person mit den geforderten technischen und sozialen Fähigkeiten vorhanden. Wie das agile Manifest [Bec01] schon schreibt: Die besten Architekturen, Anforderungen und Entwürfe entstehen durch selbstorganisierte Teams.

Kasten 3: Rolle oder Person?

zusätzlich zum Pair Programming konkrete Workshops durchführen, um das Team für diese Themen zu sensibilisieren und gemeinsame Lösungen zu finden. Ebenfalls können Teamregeln bezüglich Architektur und Design zusammen erarbeitet werden.

Unterstützung des Scrum Masters
Der agile Softwarearchitekt unterstützt den Scrum Master dabei, effizientere und effektivere Wege zur Softwareentwicklung zu finden. Dies ist besonders wichtig, wenn der Scrum Master keinen großen technischen Hintergrund hat.

Was einen agilen Softwarearchitekten auszeichnet

Ein agiler Softwarearchitekt zeichnet sich vor allem durch die folgenden Eigenschaften aus:

Kommunizieren
Der agile Softwarearchitekt kommuniziert mit allen Beteiligten, den Endbenutzern, dem Sponsor, dem Betrieb und den Entwicklern (siehe Abbildung 2). Anders als in traditionellen Projekten findet diese Kommunikation von Gesicht zu Gesicht statt und nicht über Dokumente, wie Anforderungs- und Softwarearchitektur-Dokumente.

Effektive Tools einsetzen
An die Stelle von UML-Werkzeugen und -Dokumenten treten für die direkte Kommunikation Whiteboard, Sketches, Prototypen, Stift und Papier. Diese Werkzeuge eignen sich wesentlich besser zur Vermittlung von Ideen (siehe Kasten 2).

Teil des Teams
Der agile Softwarearchitekt sitzt nicht in seinem Elfenbeinturm, sondern mitten im Team. Er ist Teil des Teams und in die Planung, Schätzung, Umsetzung und Wartung des Produkts involviert (siehe Kasten 3).

Lernen
Als technologischer Anführer muss sich der agile Softwarearchitekt kontinuierlich weiterbilden. Nur durch das Verständnis der neuen Technologien, *State-of-the-Art*-Designfähigkeiten und -Programmiertechniken kann eine Architektur entstehen, die die Risiken und Aufwände minimiert und gleichzeitig die Flexibilität maximiert.

Entscheiden
Zu guter Letzt ist es die Hauptaufgabe des agilen Softwarearchitekten zu entscheiden. Bei einem agilen Vorgehen sind die Entwicklungszyklen so kurz, dass das Verzögern von Entscheiden das ganze Team zum Stillstand zwingen kann. Der Architekt bringt die Beteiligten dazu, dass Entscheidungen gefällt werden, auch wenn diese eventuell in Zukunft wieder abgeändert werden müssen. Doch das ist das kleinere Problem, denn eine agile Architektur erlaubt Änderungen auch spät im Projekt. ■

Literatur & Links

[Bec01] K. Beck et. al, Prinzipien des agilen Manifests, 2001, siehe: http://agilemanifesto.org/iso/de/principles.html

[Hru12] P. Hruschka, G Starke, Knigge für Softwarearchitekten, entwickler.press, 2012

[Hru13] P. Hruschka, G. Starke, arc42 – Resourcen für Software-Architekten, 2013, siehe: http://www.arc42.de/

[Sca] Scaled Agile Framework, siehe: http://scaledagileframework.com