Software- und System-Architekturen richtig bewerten

30 August 2014 | **Insight Zühlke** | **Matthias Kraaz**, **Fabienne Lorenz**
Lesezeit: 11 Minutes

Die richtige Architektur ist ein Schlüsselfaktor für softwareintensive Systeme. Sie wirkt sich direkt auf Zeit und Kosten für die Implementierung und Pflege sowie auf die Leistungsfähigkeit des Systems aus. Eine schlecht passende Software- oder System-Architektur birgt das Risiko, viele Mannjahre an Entwicklungszeit zu verpulvern. Um das zu verhindern, bewerten wir bei Zühlke systematisch die Architektur von geplanten oder von existierenden Systemen. Bei den geplanten Systemen sollen durch die Bewertung die Projektrisiken gesenkt werden. Existierende Systeme werden in der Regel dann einer gründlichen Bewertung unterzogen, wenn Probleme auftauchen oder eine Weiterentwicklung ansteht.

Dazu haben wir eine Methode entwickelt: das **Zühlke Architecture Assessment Framework**. Ablauf und Methode beruhen auf der international anerkannten Architecture Tradeoff Analysis Method (ATAM^{SM}). Ergänzt wird das Vorgehensmodell des ZAAF durch eine fortlaufend weiterentwickelte Sammlung von Guidelines, Checklisten und Vorlagen.

Ein Assessment dauert maximal einige Wochen und hat typischerweise folgenden Ablauf:

- Festlegung der Ziele und des Gegenstands des Assessments mit dem Sponsor
- Kickoff-Meeting mit den Architekten und Stakeholdern
- Architektur-zentrierter Workshop
- Stakeholder-zentrierter Workshop
- Präsentation der Ergebnisse

Durchgeführt wird das Assessment von unseren erfahrenen Architekten, die schon vielmals ähnliche Software und Systeme entwickelt haben. Den Ablauf eines Assessments skizziere ich hier anhand zweier fiktiver Architekturen. Beide Male handelt es sich um ein bereits bestehendes Produkt für die Aufbewahrung von Nüssen mit Internet-Anbindung, Cloud-basiertem Backend und einer Smartphone App. Die **A-Hörnchen** haben – wie sich zeigen wird – eine mustergültige Architektur abgeliefert. Die **B-Hörnchen** haben zum Zwecke dieses Artikels alles falsch gemacht, was man nur falsch machen kann.

Ziele der Architektur-Assessments

Ziel und Gegenstand des Assessments werden vom Sponsor festgelegt. Bei einer komplexen Systemlandschaft oder einem großen Produktportfolio ist eine klare Abgrenzung des

Gegenstands der Untersuchung immens wichtig, um die richtigen Personen bei den folgenden Schritten einzubinden und eine Fokussierung auf die relevanten Themen zu gewährleisten.

Bei den **A-Hörnchen** wird das Assessment von einem Kapitalgeber beauftragt, der eine größere Investition in das junge Unternehmen absichern möchte. Der Blickwinkel des Assessments soll möglichst breit sein, um alle Risiken aufzudecken. Das gesamte System soll betrachtet werden und auf alle relevanten Qualitätsattribute geprüft werden. Die Ankündigung einer hervorragenden Dokumentation von Anforderungen und Architektur stimmt die Gutachter zuversichtlich, das Assessment in der vorgegebenen Zeit durchführen zu können.

Bei den **B-Hörnchen** wird das Assessment vom Leiter der Entwicklung beauftragt. Er ist generell um die Qualität der Software besorgt. Aktuell liegt er im Clinch mit dem Chef der Produktion. Der meint, die Software mache ihm absichtlich das Leben schwer. Prinzipiell soll auch das Gesamtsystem betrachtet werden. Am Ende sollte außerdem eine Antwort auf die Kritik aus der Produktion stehen.

Kickoff Meeting

Das Assessment beginnt mit einer Präsentation der Geschäftsziele und der Motivation für das Assessment. Leiter ist meist der Sponsor. Im Anschluss stellen wir den beteiligten Architekten und Stakeholdern die ZAAF-Methode vor. Als Hausaufgabe bekommen die Architekten die Vorbereitung der Architekturpräsentation im ersten architekturzentrierten Workshop.

Die **A-Hörnchen** hören bei der Präsentation der Geschäftsziele nichts wirklich Neues, freuen sich aber über eine vollständige, priorisierte Liste der Geschäftsziele. Die Vorbereitung des architekturzentrierten Workshops ist eine Sache von Minuten, da die benötigten Informationen allesamt gut dokumentiert vorliegen.

Die **B-Hörnchen** sind über die Geschäftsziele baff erstaunt, haben sie sich doch mit den „kommerziellen“ Aspekten des Systems noch nie beschäftigt. Für die Hausaufgaben planen sie mehrere Tage ein, müssen doch die meisten Informationen erst mal aus vielen Köpfen gesammelt und in eine präsentierbare Form gebracht werden.

Architekturzentrierter Workshop

Am architekturzentrierten Workshop nehmen nur die Gutachter und die Architekten teil. Der Workshop beginnt mit einer Präsentation der architekturtreibenden Anforderungen und der Architektur durch die Architekten. Die Gutachter haben nach Möglichkeit die Architekturdokumentation bereits vorweg erhalten und studieren

können. Architekturtreibende Anforderungen sind meistens nicht-funktional und beschreiben Qualitätsattribute wie Wartbarkeit, Effizienz, Zuverlässigkeit. Die nicht-funktionalen Anforderungen an eine Architektur sind manchmal sehr pauschal formuliert, beispielsweise „Die Software soll erweiterbar sein.“ Als Basis für eine Analyse reicht das nicht aus. Szenarien dienen der Konkretisierung anhand eines Beispiels. Ein **Szenario** enthält folgende Informationen:

- **Umgebung**, in der das Szenario stattfindet
- **Auslöser**, auf den das System reagieren muss
- Erwartete **Reaktion** des Systems
  Szenarien sollten so konkret wie möglich sein und die Reaktion idealerweise messbar beziehungsweise testbar sein.

  Die **A-Hörnchen** haben erfolgreich eingefordert, dass die Qualitätsanforderungen messbar sind oder wenigstens durch Beispiele konkretisiert werden. Praktisch sind alle Informationen schon vorhanden und die Liste der Szenarien kann in Windeseile erstellt werden. Bei der Umgebung unterscheiden die Architekten zwischen Entwicklung, Test, Wartung, Normalbetrieb und degradierten Betrieb ohne Internetzugang. Aus den Beispielen zur Konkretisierung der Anforderungen ergeben sich direkt Auslöser und gewünschte Reaktion:

- Umgebung: **Entwicklung**. Auslöser: Die zugrundeliegende Datenbank soll ausgetauscht werden. Reaktion: Die Datenbank kann durch Anpassung der Konfiguration ersetzt werden
- Umgebung: **Test**. Auslöser: Verhalten beim Ausfall der Internetverbindung soll getestet werden. Reaktion: Über eine Testschnittstelle kann das Ereignis simuliert werden.
- Umgebung: **Normalbetrieb**. Auslöser: Die maximale Last wird erreicht. Reaktion: Keinerlei Transaktionen gehen verloren
  Die **B-Hörnchen** haben ihre Architektur auf diffuse Qualitätsanforderungen aufgebaut. Diese Anforderungen werden teilweise zum ersten Mal dokumentiert. Bei einem Teil der Anforderungen gibt es nur Vermutungen, welche Wünsche der Stakeholder hat. Es finden sich nur Szenarien für den Normalbetrieb.

  Listen von **Qualitätsattributen**, wie man sie in der ISO/IEC 9126 oder der neueren ISO/IEC 25010 findet, helfen bei der Identifizierung und Kategorisierung der nicht-funktionalen Anforderungen und der zugehörigen Szenarien. Nicht jedes Qualitätsattribut ist für jede Software wichtig und so wird man auch nicht zu jedem Qualitätsattribut ein Szenario finden.

> Umgekehrt können für eine Software aber durchaus Qualitätsattribute relevant sein, die man so in keiner standardisierten Liste wiederfindet. Die relevanten Qualitätsattribute, nicht-funktionalen Anforderungen und Szenarien ergeben den sogenannten **Utility Tree**, eine übersichtliche, hierarchische Darstellung.

Die **A-Hörnchen** haben bereits beim Architekturentwurf die Relevanz von Qualitätsattributen untersucht und zaubern praktisch den fertigen Utility Tree aus dem Hut.

Die wesentlichen Qualitätsattribute sind Wartbarkeit, Effizienz und Sicherheit. Auch die abermalige Überprüfung fördert keinerlei Qualitätsattribute zu Tage, die unzureichend betrachtet wurden.

Die **B-Hörnchen** schlagen sich tapfer und kommen auf Effizienz und Sicherheit als relevante Qualitätsattribute. Wegen der bekannten Probleme bei der Produktion wird Produzierbarkeit mit aufgenommen, den Architekten sind aber keine diesbezüglichen Anforderungen bekannt.

> Eine Software-Architektur ist in der Regel die Manifestation einer Menge von bewusst und unbewusst getroffenen Entscheidungen, die nur teilweise dokumentiert sind.

Basierend auf der Präsentation der Architektur und der vorweg erhaltenen Dokumentation identifizieren die Gutachter die Architekturentscheidungen und machen Software-Architektur auf diese Weise konkret und bewertbar. Sämtliche Architekturentscheidungen werden nach folgenden Kriterien analysiert:

- Positive Wirkung von Architekturentscheidungen (**non-risk**): Jede Architekturentscheidung sollte der Befriedigung eines Qualitätsattributs, einer Anforderung oder eines Szenarios dienen.
- Mögliche negative Konsequenzen von Architekturentscheidung (**risk**)
- Für den Erfolg der Architektur essentielle Annahmen oder Eigenschaften (**sensitivity point**)
- Zielkonflikte zwischen architekturtreibenden Anforderungen (**trade-offs**): Nicht-funktionale Anforderungen stehen regelmäßig im Zielkonflikt zueinander. Eine Architekturentscheidung kann daher simultan mehrere Qualitätsattribute positiv und negativ beeinflussen.
  Die **A-Hörnchen** haben wiederum den ZAAF fast schon beim Architekturentwurf fertig gehabt. Sämtliche Architekturentscheidungen und die zugrundeliegenden Annahmen sind gut und anschaulich dokumentiert und wohlbegründet mit einer Abwägung zwischen verschiedenen Anforderungen. Insbesondere die Anforderungen an die Sicherheit wurden sorgfältig mit den Bedürfnissen während der Entwicklung und der Produktion abgestimmt.

  Den **B-Hörnchen** geht ein Licht auf, wie so etwas abstraktes wie eine Software-Architektur

eigentlich bewertet werden kann, als die Gutachter anfangen, verschiedene Design-Entscheidungen als Architekturentscheidungen zu identifizieren. Mit vielen Fragen nach dem warum und wieso kann auch aus der Beschaffenheit der Software zurück auf die Architekturentscheidungen geschlossen werden. Die meisten Architekturentscheidungen können dem Bereich Sicherheit zugeordnet werden, während keinerlei Architekturentscheidungen zur Unterstützung der Wartbarkeit identifizierbar sind. Die Analyse wird nach dem Workshop von den Gutachtern fortgesetzt. Bei Bedarf werden weitere Informationen von den Architekten eingeholt. Aus den vervollständigten und konsolidierten Analyseergebnissen entsteht eine erste Version des Assessment-Berichts.

Code-Inspektion

Bei einer bereits realisierten Software-Architektur kann eine nachfolgende oder begleitende Code-Inspektion wertvolle Hinweise auf Defizite in der Umsetzung der Software-Architektur liefern. Die Code-Inspektion deckt die Ursache für Probleme auf, die nicht in der Software-Architektur selbst begründet sind.

Bei den **A-Hörnchen** tritt nichts Überraschendes zu Tage. Sie haben während der Entwicklung peinlich genau darauf geachtet, dass die entworfene Architektur auch so umgesetzt wird. Zu Anfang der Entwicklung wurde ein Skelett der Software erstellt, das die Tragfähigkeit der Architektur bewies und nur noch mit Features ausgefleischt werden musste. Die Erkenntnisse daraus sind in den Architekturentwurf zurückgeflossen und sauber dokumentiert worden. Während der gesamten Entwicklung wurde die Einhaltung von Vorgaben zum Design einzelner Komponenten und zur Code-Qualität von den Architekten überwacht.

Die **B-Hörnchen** müssen feststellen, dass die Architektur nur teilweise wie beabsichtigt umgesetzt wurde. Die Entwickler verteidigen sich mit der unklaren Dokumentation und Kommunikation der Architektur. Die Gutachter dokumentieren die Deltas zwischen Soll- und Istzustand.

Stakeholder-zentrierter Workshop

Beim Stakeholder-zentrierten Workshop nehmen sowohl die Architekten als auch die Stakeholder teil. Der Sponsor kann selbst als Stakeholder auftreten oder die Wahrnehmung seiner Interessen durch die übrigen Stakeholder wahrnehmen lassen.

Ziel des Workshops ist die Ergänzung der von den Architekten identifizierten Szenarien durch die Stakeholder und den Aufbau eines gemeinsamen Verständnisses zwischen Stakeholdern und Architekten über die Software- beziehungsweise System-Architektur.

Der stakeholderzentrierte Workshop beginnt mit einer kondensierten Präsentation der Ergebnisse des Architektur-zentrierten Workshops, damit die Stakeholder auf den bereits gewonnenen Erkenntnissen aufbauen können. Aus den vielen Anforderungen der Stakeholder müssen die **architekturtreibenden Anforderungen** herausgefiltert werden. Dies sind Anforderungen, die zu grundlegenden Entscheidungen über die realisierte Lösung führen – nicht Anforderungen, die sich durch eine lokale, taktische Designentscheidung mit begrenzter Auswirkung innerhalb einer Software-Komponente lösen lassen.

Die Anforderungen der Stakeholder werden wiederum in die Szenario-Form gebracht, um die Analyse durch die Architekten und Gutachter zu ermöglichen. Bei der Sammlung von Szenarien kann man folgende Typen von Szenarien unterscheiden:

- **Use Case Scenario**: Ein ganz gewöhnlicher Fall der Benutzung der Software
- **Growth Scenario**: Wie genügt die Architektur steigenden Anforderungen?
- **Exploratory Scenario**: Eine extreme Belastung der Architektur

Sämtliche Typen von Szenarien sind wertvoll, um die Stärken und die Grenzen einer Architektur zu erforschen und herauszustellen. Den Architekten werden somit die Anforderungen seitens der Stakeholder klarer und den Stakeholder die Eigenschaften der Architektur. Aus der Fülle von Szenarien können aber nur die wichtigsten direkt im Workshop analysiert werden. Die Analyse wird wiederum nach dem Workshop von den Gutachtern fortgesetzt und die finale Version des Berichts entsteht in Rücksprache mit den Teilnehmern und dem Sponsor des Assessments.

Die **A-Hörnchen** haben den gesamten Lebenszyklus des Produkts und der umgebenden Infrastruktur betrachtet, um wirklich alle Stakeholder zu berücksichtigen: also auch Hardware-Entwicklung, Software-Entwicklung, Testabteilung, Produktion, Betrieb, Service und so fort. Weitere Stakeholder konnten auch die Gutachter nicht identifizieren.

Die Architektur hält vielen explorativen Szenarien nicht stand, was aber der Erwartung der Architekten entspricht. Den Architekten war bewusst, dass sie nicht jede noch so visionäre Idee abdecken können, sondern sich teilweise auf den Business Case zurückziehen müssen, um mit angemessenen Aufwand zum Ziel zu kommen.

Diese Balance haben Sie in Zusammenarbeit mit den Stakeholdern hergestellt. Die reguläre Geschäftsentwicklung ist hingegen durch die durchweg befriedigten Growth Szenarien abgedeckt.

Die **B-Hörnchen** haben bei ihrem Architektur-Entwurf lediglich die typischen Endnutzer, vertreten durch das Produktmanagement, berücksichtigt. Die Gutachter sind daraufhin auf

die Suche nach den übrigen Stakeholdern gegangen und haben diese eingeladen. Die B-Hörnchen sind etwas überrascht über die vielen unbekannten Gesichter, insbesondere den schlecht gelaunten Chef der Produktion.

Von letzterem werden sie mit dem Szenario der effizienten Produktion der Aufbewahrungsbox konfrontiert. Derzeit sind zahlreiche manuelle Schritte und viel Zeit für das Deployment der Software vonnöten – und das bei jedem Gerät! Was die Architekten sich denn dabei gedacht hätten?

Aufgrund der Vorarbeit im architekturzentrierten Workshop können die Architekten darstellen, wie die verschiedenen Sicherheitsmaßnahmen in der Architektur die Vereinfachung der Produktion verhindern. Aber auch die Anforderungen an die Sicherheit werden durch die Szenarien der Stakeholder bestätigt. Der Chef der Produktion ist etwas getröstet, dass das Vorgehen der Architekten einen anderen Zweck hatte, als ihm das Leben schwer zu machen. Die Abstimmung der Prioritäten der Szenarien ergibt jedoch, dass die Maßnahmen zur Sicherheit etwas übertrieben sind und die Produktion durch geringfügige Änderungen in der Architektur wesentlich effizienter gestaltet werden kann.

Abschluss des Architektur-Assessments

Das Assessment wird mit einer Präsentation der Befunde und der daraus abgeleiteten Handlungsempfehlungen abgeschlossen.

Die **A-Hörnchen** sind erleichtert, dass keine wesentlichen Schwächen aufgedeckt wurden.

Die **B-Hörnchen** und ihre Stakeholder freuen sich über das gewonnene gemeinsame Verständnis bezüglich der Architekturanforderungen und der derzeitigen Architektur und geloben bessere Zusammenarbeit in der Zukunft. Diese Zusammenarbeit beginnt gleich anschließend mit der schrittweisen Umsetzung der Handlungsempfehlungen.

May contain nuts. No animals were harmed in the production of this article.