



» **Entwicklung** » **Methoden**

Daniel Takai

26. September 2017

Entscheidungswege in der Softwarearchitektur identifizieren



© djama / Fotolia.com

Entscheidungen in einem Softwareprojekt sind oft wegweisend, weil sie sich später in vielen Fällen nicht mehr rückgängig machen lassen. Beispielsweise sind Entscheidungen für eine konkrete Technologie – etwa eine Programmiersprache oder ein Framework – stets mit einem hohem Schulungs- und Implementierungsaufwand und damit einer substantiellen Investition

verbunden. Eine wesentliche Aufgabe des Architekten ist, diese Investitionen zu schützen, und zwar durch die strukturierte und disziplinierte Identifizierung, Vorbereitung und Herbeiführung von Entscheidungen. Um dies tun zu können, müssen wir verstehen, welche Arten von Entscheidungen in der Architektur getroffen werden.

Paul Clements hat die Architektur als Schlüssel zum Verständnis eines Systems bezeichnet. Ihre Aufgabe sei, eine hoch-informative Kommunikation zwischen den Akteuren und Stakeholdern herzustellen, die sonst keine gemeinsame Sprache über das System hätten [1]. Der Architekt ist an dieser Kommunikation interessiert, weil er das Zusammenspiel der Bedürfnisse der Akteure mit dem Lösungsraum bewerten muss, welcher durch Entscheidungen immer weiter verfeinert wird. Um erfolgreich, aber auch effizient arbeiten zu können, muss er die Entscheidungen also nicht nur identifizieren, sondern dabei auch die wesentlichen und wichtigen von den unwesentlichen und unwichtigen Entscheidungen trennen können.

Das Prinzip des *Last Responsible Moment* (LRM) besagt, dass Entscheidungen erst dann gefällt werden müssen, wenn es auch wirklich notwendig ist, die Entscheidung zu treffen [2]. Haben Sie beispielsweise entschieden, das Log-Management an einen SaaS-Dienst zu delegieren, so müssen Sie sich nicht für einen konkreten Service entscheiden, bevor dieser für eine Test- oder Schulungsumgebung das erste Mal benötigt wird. Die Klassifikation von Entscheidungen nach Dringlichkeit kann demnach helfen, die Planung zu verbessern.

Um Entscheidungen identifizieren zu können, ist es hilfreich, eine Unterteilung der möglichen Entscheidungen nach einem Schema vorzunehmen. Eine mögliche Typisierung von Entscheidungen in der Architektur ist die Unterteilung in drei Kategorien: Domäne, Technologie sowie Kultur und Politik.

Domänenentscheidungen

Die Domänenentscheidungen betreffen das Geschäft der Zielorganisation, sind also in hohem Maße vom konkreten Kontext abhängig, beispielsweise der Branche. Ziel eines Systems sind in der Regel Domänenziele, beispielsweise die Steigerung der Effizienz von Arbeitsprozessen, die Verbesserung der Qualität im Kundendienst oder ein höherer Abverkauf im Retail. Die Domäne ist die Arbeit einer Organisation in ihrer Umwelt und damit sehr konkret und spezifisch. Durch Techniken des domänengetriebenen Entwurfs kann sie durch Domänenmodelle dargestellt und kommuniziert werden [3]. Der domänengetriebene Entwurf ist eine fachliche Dekompositionstechnik. Ein möglicher Oberbegriff über das Wesen der Domäne ist die sogenannte Geschäftsarchitektur.

Wesentliche Entscheidungen, die bei der Anwendung dieser Technik getroffen werden müssen, sind der Schnitt der Domänenmodelle sowie die Kollaboration der Domänen

Autor



Daniel Takai

Daniel Takai ist Enterprise-Archit und Mitgründer der Silberrücken, einer Boutique Consulting Firma, spezialisiert auf die Digitale...

>> [Weiterlesen](#)

Publikationen

[Architektur für Websysteme](#)

Newsletter

Unser Newsletter informiert Sie regelmäßig und kostenlos über Neuigkeiten, Artikel und Veranstaltungen zu aktuellen Themen.



Nachrichten

untereinander. Entscheidungen hierüber benötigen also ein tiefes Verständnis des jeweiligen Business. Wenn Entwickler und Architekten nicht zur Zielorganisation gehören, dann fehlt dieses Wissen automatisch, weil der konkrete Kontext unbekannt ist. In diesem Fall ist also eine Ergänzung des Teams um Fachexperten notwendig. Häufig kommt die Rolle des Fachexperten dem Business Analyst oder Requirements Engineer zu.

Entscheidungen in der Domäne sind häufig mit dem Geschäftsmodell der Zielorganisation verbunden. Entsprechend kann es sich lohnen, sichtbar zu machen, inwieweit das System das Geschäftsmodell verändern könnte, oder umgekehrt. Beispielsweise ist die Preisberechnung im Retail sowohl im B2C als auch im B2B eine komplexe und schwierige Domäne, die von vielen anderen Domänen benötigt wird. Häufig ist dann eine Implementierung als dedizierter und isolierter Service eine gute Wahl. Gar keine gute Wahl ist es, die Abhängigkeit zum Geschäftsmodell nicht zu besprechen.

“ Ein System wird im Erleben seiner Benutzer durch seine Qualitäten geprägt.

Die Reflexion der Entscheidungen in der Domäne ist für den Architekten wichtig, weil diese Auswirkungen auf den Entwurf haben können. Aus ihnen ergibt sich zudem die Wirtschaftlichkeit eines Systems, sodass der Architekt durch seine Arbeit die Lebensdauer mitbestimmen kann: Ein wirtschaftliches System läuft länger.

Technologie

In einem Softwareprojekt gibt es außerdem technologische Entscheidungen, welche häufig durch Rahmenbedingungen gegeben sind. Eine Rahmenbedingung ist eine Anforderung, die nicht vom Team verändert werden kann. Beispielsweise kann es eine Richtlinie geben, die die Verwendung einer bestimmten Technologie vorgibt, etwa die Datenbank. Daran lässt sich erst einmal nichts ändern, aber man kann reflektieren, dass diese Entscheidung vorgegeben wurde, und das auch kommunizieren. Eine solche Rahmenbedingung kann sich durchaus auch negativ auf ein Softwareprojekt auswirken. Auf diese Weise kann die vorgegebene Entscheidung sichtbar gemacht und gespiegelt werden. Wenn dann die Rahmenbedingung geändert werden kann, hat der Architekt sehr gut gearbeitet. Kann die Bedingung nicht geändert werden und beeinflusst sie das Projekt tatsächlich negativ, so hat man wenigstens eine Entschuldigung.

Ein System wird im Erleben seiner Benutzer durch seine Qualitäten geprägt, beispielsweise wie schnell es antwortet, wie gut es aussieht oder ob es den Eindruck von Sicherheit vermittelt. Die strukturierte Planung der gewünschten Qualitäten kann durch ein Qualitätsmodell geschehen, das die Qualitäten beschreibt, auf deren Grundlage bewertet werden soll. Ein Qualitätsmodell beschreibt, bewertet und sagt Qualität voraus. Für die Planung sollte also ein Qualitätsmodell verwendet werden, das zum geplanten Einsatz des Systems passt.

Das bekannteste Qualitätsmodell für Software ist ISO 25010, ein generisches Qualitätsmodell, das für alle Softwaresysteme gelten soll. Deswegen ist dieses Modell schwierig gegen ein konkretes Geschäftssystem bewertbar. Es gibt abhängig vom Einsatzgebiet viele verschiedene Arten von Software. Transaktionsmonitore im internationalen Zahlungsverkehr oder die Steuerungssoftware einer Drohne haben andere Qualitäten aufzuweisen als ein Hotelbuchungsdienst.

Da ich ausschließlich an sozio-technischen Geschäftssystemen arbeite, verwende ich stets ein eigenes Modell als Grundlage, das zu dieser Art von Systemen passt und das ich dann situativ modifiziere. Zu den wesentlichen Qualitäten gehören hier die Wartbarkeit, die Performance, die Zuverlässigkeit und die Informationssicherheit. Zur Erlangung der geplanten Qualitäten gilt es dann, die richtigen technischen Entscheidungen zu treffen, beispielsweise Continuous Deployment zur Verbesserung der Wartbarkeit bzw. der Änderbarkeit einzusetzen oder eben nicht. Bei Websystemen ist eine zentrale Entscheidung beispielsweise die der Skalierbarkeit, die den Lösungsraum stark einschränkt. Die Skalierbarkeit ist ein sogenanntes Submerkmal der Performance.

Es gibt viele Qualitäten, die sich als Grundlage für die Identifizierung der notwendigen Entscheidungen eignen [4]. Hier sollen drei Varianten unterschieden werden.

- Ein **Cloud-Native-System** ist ein System, das in einer Cloud betrieben werden soll. Solch ein System sollte sowohl zuverlässig als auch elastisch sein, was bedeutet, dass das System seine Kapazität automatisch anpassen kann und zudem auch beim Auftreten von Störungen seinen Dienst nicht versagt.

22.11.2022 Einführung in CI/CD und DevOps mit Gitlab

In diesem kostenfreien Workshop erhältst Du in Form eines hands training eine Einführung in Conti Integration (CI) und Continuous.

>> [Weiterlesen](#)

27.10.2022 Unternehmen berichten live aus ihrem Multiprojektmanagement!

Melden Sie sich jetzt für das Onl Forum am 17. November von 10 13:00 Uhr an! Blicken Sie mit ur gemeinsam auf dem Blue Ant Fo

>> [Weiterlesen](#)

25.10.2022 Deutsche IT-Security-Konferenz 2022: IT-Security-Tage, die Konferenz des Fachmagazi Informatik

Die IT-Security-Tage finden remc Rahmen der IT-Tage statt. Das Besondere daran: Du kannst nic von jedem Ort teilnehmen, sond

>> [Weiterlesen](#)



- Ein **Reaktives System** ist ein nachrichtengesteuertes Cloud-Native-System, bei dem die Services untereinander asynchron Nachrichten austauschen und deswegen entkoppelt und isoliert ihre Arbeit verrichten können.
- Ein **Geschäftssystem** hat neben den Qualitäten der Zuverlässigkeit und der Performance auch noch Aspekte der Informationssicherheit und Wartbarkeit. Ein Geschäftssystem kann Cloud-Native oder Reaktiv sein, muss es aber nicht.

Für den Architekten ist die quantifizierbare Bewertung über Qualitätsmodell eine Hilfe bei der strukturierten Entscheidungsfindung. Bei vielen Systemen findet diese Bewertung übrigens meiner Erfahrung nach nicht statt, sodass es immer wieder zu Überraschungen kommt, wenn das System am Ende wichtige Stakeholder enttäuscht. Dabei kann ein Architekt die wesentlichen Punkte mit vertretbarem Aufwand identifizieren und sichtbar machen, was am Ende des Tages Geld, Zeit und Nerven spart. Außerdem sollten auch neue Technologien zum Einsatz kommen, aber man tut sich schwer damit, hierfür einen guten Grund zu finden. Die Reflexion über die Qualitäten eines Systems ist dann eine Hilfe, um solche Gründe finden zu können.

Kultur und Politik

Nicht überraschend ist die dritte Kategorie von Entscheidungen kultureller und politischer Natur. Sozio-technische Systeme haben eben auch eine soziale Seite mit Individuen und deren eigenen Hintergründen, Zielen und Motiven, die einen großen Einfluss auf ein Softwareprojekt haben. Die Frage nach Entscheidungen in diesem Bereich beginnt mit der Organisationsstruktur, denn wer wann wo sitzt und verfügbar ist, beeinflusst die Zusammenarbeit stark. Hier spielen auch wieder Rahmenbedingungen hinein, denn wenn ein Fachexperte noch auf anderen Projekten arbeiten muss, dann ist er eben nur am Dienstag da, um Fragen zu beantworten. Besonders gut funktioniert die Zusammenarbeit, wenn man die gleiche Sprache spricht, denn Missverständnisse sind sonst vorprogrammiert bzw. im Extremfall ist Kommunikation gar nicht möglich.

“ Das Ziel ist erreicht, wenn alle im Team die Entscheidungen nachvollziehen können und sich gut informiert fühlen.

Für den Architekten ist es hilfreich, die Ziele und Motivationen der Akteure zu verstehen, um Synergien unterstreichen zu können. Als Hilfsmittel gibt es hier die Modellierungssprache Archimate für Unternehmensarchitekturen. Sie erlaubt die strukturierte Modellierung der Akteure, ihrer Werte und Ziele ebenso wie ihrer Prinzipien und Werte [5]. Da es sich bei diesen Zielen und Motiven um geschäftliche Dinge handelt, die nicht privat sind, ist eine offene Diskussion vertretbar. Tatsächlich kommunizieren viele Stakeholder diese Dinge sehr gerne, wenn sie in die Entscheidungsfindung integriert werden.

Die Integration der Stakeholder in die Entscheidungsprozesse ist also ein wichtiger Erfolgsfaktor. Das Ziel ist erreicht, wenn alle im Team die Entscheidungen nachvollziehen können und sich gut informiert fühlen. Außerdem sollte sich niemand fragen müssen, warum seine Meinung nicht gehört wurde, obwohl sie doch so offensichtlichen Einfluss gehabt hätte. Hierbei muss man sich immer wieder vor Augen halten, dass Software unsichtbar ist und erst durch aktive Kommunikation für die anderen fassbar wird.

Fazit

Ich habe drei Kategorien von Entscheidungen beschrieben, deren Reflexion und Kommunikation für ein Softwareprojekt hilfreich sind. Allen drei Typen von Entscheidungen ist eigen, dass sie zunächst identifiziert werden müssen, und es benötigt Führung, um sie identifizieren zu dürfen. Bei den technologischen Entscheidungen hilft eine Bewertung mit einem Qualitätsmodell als Grundlage. Viele Entscheidungen sind durch Rahmenbedingungen eingeschränkt, die sichtbar gemacht werden wollen. Gar nicht in diesem Artikel beschrieben wurden Risiken, die natürlich eng mit Entscheidungen in Beziehung stehen.

Allen Entscheidungen ist zudem eigen, dass sie eine hohe Kompetenz des Architekten in der Kommunikation benötigen. Bei sozio-technischen Systemen ist das gegenseitige Verständnis die wichtigste Voraussetzung für den gemeinsamen Erfolg.

Quellen

- [1] P. Clements; R. Kazman; M. Klein: Evaluating Software Architecture. Addison-Wesley, 2002 (SEI Series in Software Engineering)
- [2] M. Poppendiek: Lean Software Development: An Agile Toolkit for Software Development Managers. Addison-Wesley, 2003
- [3] E. J. Evans: Domain-Driven Design: Tackling Complexity in the Heart of Software. Addison-Wesley, 2003

[4] D. Takai: Architektur für Websysteme: Serviceorientierte Architektur, Microservices, Domänengetriebener Entwurf, Hanser Verlag, 2017

[5] Archimate: [Dokumentation](#)



Autor



Daniel Takai

Daniel Takai ist Enterprise-Architekt und Mitgründer der Silberrücken AG, einer Boutique Consulting Firma, spezialisiert auf die Digitale Transformation von Unternehmen und Non-Profits in der Schweiz.

[>> Weiterlesen](#)

Publikationen

- [Architektur für Websysteme](#)

Das könnte Sie auch interessieren



Wie DevOps zum Stahlwerk kam

Sponsored Post



Mit Domain Storytelling ein Minimum Viable Product entwickeln



Eclipse Theia: die nächste Generation der Eclipse-Plattform



Revisionssicher testen mit Cucumber und Testcontainers



End-2-End-Tests mit Playwright-Test



Hybride Apps – die All-In-One-Lösung für Frontends

Kommentare (0)

Neuen Kommentar schreiben

Name:

E-Mail-Adresse:

Kommentar:

Ich verstehe und akzeptiere die **Datenschutzbestimmungen**.

Absenden

Newsletter

- ▶ *Aktuelle Artikel*
- ▶ *Aktuelle Nachrichten*
- ▶ *Aktuelle Konferenztermine*

 Newsletter abonnieren

Kategorien

- Startseite
- Management
- Entwicklung
- Betrieb

Service

- Aktuelle Meldungen
- Konferenzkalender
- IT-Jobs
- Sitemap

Information

- Impressum
- Über Uns
- Media
- Datenschutz
- Nutzungsbedingungen
- Kontakt

Folgen Sie uns



© Alkmene Verlag GmbH 2022