1. Termin : *Systematischer* Softwarearchitekturentwurf

Dauer: 4 Einheiten

Diskussionsthemen

- Was ist Softwarearchitektur? Architektur vs. Design
- Anforderungen an eine SW-Architektur?
- Erstellung einer SW-Architektur (Vorgehensweise, Voraussetzungen, Anforderungen, iterativ, best practice?)
- Weiterentwicklung und Lebenszyklus einer SW-Architektur, spezielle Anforderungen hierfür?
- Stakeholder

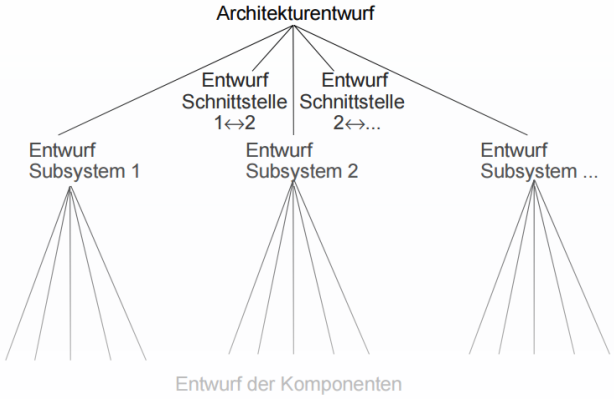
Adaptiver Softwareentwickelungsprozess : Vom Entwurf zur Implementierung

Agile und adaptive Softwareentwicklung bedeutet, den Entwicklungsprozess flexibler und schlanker zu machen, als das bei früher verbreiteten Vorgehensmodellen der Fall ist. Eine agile Vorgehensweise hat das Ziel, den Softwareentwicklungsprozess durch Abbau der Bürokratie und durch die stärkere Berücksichtigung der menschlichen Aspekte effizienter zu gestalten.
Den meisten agilen Prozessen liegt zu Grunde, dass sie versuchen, die reine Entwurfsphase auf ein Mindestmaß zu reduzieren und im adaptiven Entwicklungsprozess so früh wie möglich zu ausführbarer Software zu gelangen, die dann in regelmäßigen, kurzen Abständen dem Kunden zur gemeinsamen Abstimmung vorgelegt werden kann. Damit erfolgt die Entwicklung in einem stetigen Rückkopplungsprozess, bei dem sich die Software in jedem Iterationsschritt mehr und mehr den (Kunden-)Anforderungen annähert.

Architekturentwurf

Der Entwurf der Architektur soll einen groben Überblick geben, wie die Software aufgebaut sein wird und wie die einzelnen Softwareteile zusammenwirken, ohne jedoch auf das Design (Klassen, Methoden) oder gar Implementierungsdetails näher einzugehen (vgl. Architektur vs. Design). Im Entwurf sollten nur die grundlegendsten Entscheidungen getroffen werden, ohne zu sehr etwaige Details der Realisierung vorwegzunehmen.

Die Abgrenzung zum nachfolgenden Designentwurf ist nicht immer einfach und scharf zu ziehen, sondern vielmehr fließend. Die Grenzen hängen oft von der Größe und Komplexität des Gesamtsystems ab. Umso größer und komplexer das Gesamtsystem, umso weniger wird beim Architektur-Entwurf ins Detail gegangen, um die Übersichtlichkeit möglichst noch zu wahren.



Designentwurf

Das Design soll die zuvor geplanten Architekturvorgaben umsetzen. Eine Designplanung beschreibt bis zu jener Detailtiefe (Komponenten-, Interface-, Klassen- und Protokollebene), die eine verteilte Entwicklung (= mehrere Entwickler an verteilten Standworten) möglich macht. Der Entwicklungsfokus liegt sehr oft, wie bei der Architektur auch, auf bewährte Design- bzw. Architektur-Muster, um damit die Wiederverwendung von Softwareteilen zu forcieren.

Sollten während des Designentwurfs Anpassungen am Architekturentwurf vorgenommen werden müssen (sehr wahrscheinlich), dann wird der Entwurf schrittweise angepasst.

Implementierung

Zur Implementierung werden die einzelnen Teile der Software und die dafür entsprechenden Aufgaben im Team verteilt und die Module bzw. Komponenten entsprechend den Vorgaben implementiert.
Die Implementierung im Team kann übrigens durchaus auch ein Treiber beim Entwurf der Architektur und des Designs sein. Oft muss nicht nur der verteilte Betrieb, sondern auch die verteilte Entwicklung einer Software möglich gemacht werden. Und dann sind gerade auf Designebene exakte Beschreibungen der Schnittstellen wesentlich.

Kontinuierliche und iterative Entwicklung sich stetig ändernder Vorgaben

In agilen Softwareentwicklungsprojekten wird auf evolutionäre Softwarearchitektur gesetzt. Das heißt, bei der agilen Entwicklung erfolgt, auch durch die schon frühe Implementierung, eine wiederholte Prüfung des Entwurfs mit der Realisierung und mit den Anforderungen. Diese Rückkopplung führt zu einer kontinuierlichen Anpassung sowohl des technischen Designs als auch der Architektur an immer neue und sich stetig verändernde Anforderungen.

Demnach sind auch erst mit dem Ende der Implementierung die Architektur- und Design-Entwürfe und die Dokumentation fertig. Zuvor handelt es sich um aktuell gültige Zwischenstände der jeweiligen Entwürfe und der aktuellen Abbildung der Realisierung.
Das heißt aber nicht, dass man nun einfach drauflos programmieren sollte, und hie und da dazwischen mal dokumentiert. Je besser die Planung vorab schon ist, desto weniger grundlegende Änderungen sind nachträglich notwendig. Umso weitsichtiger geplant und entworfen wird, umso anpassungsfähiger ist letztlich auch die Software. Und umso später große Änderungen notwendig werden, umso teurer wird die Änderung letztlich. Längeres Nachdenken über mögliche Lösungsansätze schon am Beginn des Entwicklungsprozesses, kann demnach über den gesamten Lebenszyklus einer Software durchaus zu erheblichen Einsparungen bei später notwendigen Anpassungen und Erweiterungen der Software bringen.

Fazit: Wir planen nicht alles vorab bis ins letzte Detail und legen dann erst los. Das war früher mal so. Moderner Softwareentwicklungsprozess ist immer iterativ, agil, adaptiv und evolutionär. Wir nähern uns schrittweise dem fertigen Produkt und dem beweglichen Ziel, treffen zeitgerecht die wesentlichen Entscheidungen und dokumentieren in ausreichendem Umfang. Wann und in welchem Umfang nun? Naja, bei allem sollte - wie so oft im Leben - die Devise sein:

So spät/wenig wie möglich und so früh/viel wie notwendig!

Aufgabenstellung (Teamarbeit in 3er Gruppe):

Im ersten Teil dieses Labors beschäftigen wir uns mit dem systematischen Software-Architekturentwurf.

- **Hören** Sie den Podcast (jedes Teammitglied für sich):
  Episode 12 - ***Systematischer Softwarearchitekturentwurf***
  Download (mp3-Datei): http://heise.de/-353501 (35 min)

- Input zum Thema Software Architektur und die Rolle Architekt*in im Team
  How to Become a Great Software Architect • Eberhard Wolff • GOTO 2019
  https://www.youtube.com/watch?v=v_nhv6aY1Kg (43 min)

- Inputs zum Thema Adaptive Architektur
  “Agile Architecture" - Molly Dishman & Martin Fowler Keynote
  https://www.youtube.com/watch?v=VjKYO6DP3fo (38 min)

- Input zum Thema Architektur / Code
  Simon Brown on software architecture vs code
  https://www.youtube.com/watch?v=0fSRadBepYk (6 min)

- **Erarbeiten Sie den Entwurf eines verteilten, komponentenbasierten Softwaresystems** entsprechend den Anforderungen des an die Gruppe ausgegebenen fiktiven Software-produkts. Diskutieren und entwickeln Sie dabei im Team(!!) die Software-Architektur. Argumentieren Sie in der Gruppe Ihre Vorschläge und Ideen. Erarbeiten, skizzieren und beschreiben Sie gemeinsam Ihren Entwurf.

  Der Fokus sollte im ersten Schritt mal nicht im Umfang der Dokumentation, sondern vielmehr auf einer möglicherweise funktionierenden Architektur liegen. Gehen Sie davon aus, dass Sie die Architektur Ihren Kolleg:innen auch erklären und Ihre Ideen auch verteidigen können sollten.

Abgabe (Team-Abgabe):

- Was: Erster schriftlicher Entwurf der Software-Architektur

- Form: **Beschreibung Ihres ersten Architektur-Entwurfs** (als PDF-Dokument)
  immer mit Nennung von Dokumentname, Autoren, Datum, Version, Seitennummerierung (durchgehend mit Kopf-/Fußzeilen!)

  Dateityp: zip-Datei
  Dateiname: **ASA_LB24_01_***Namename1_Nachname2_Nachname3***.zip**

  Wohin: Abgabe erfolgt in Moodle
  Wann: vor dem nächsten Labortermin