Softwaretechnik

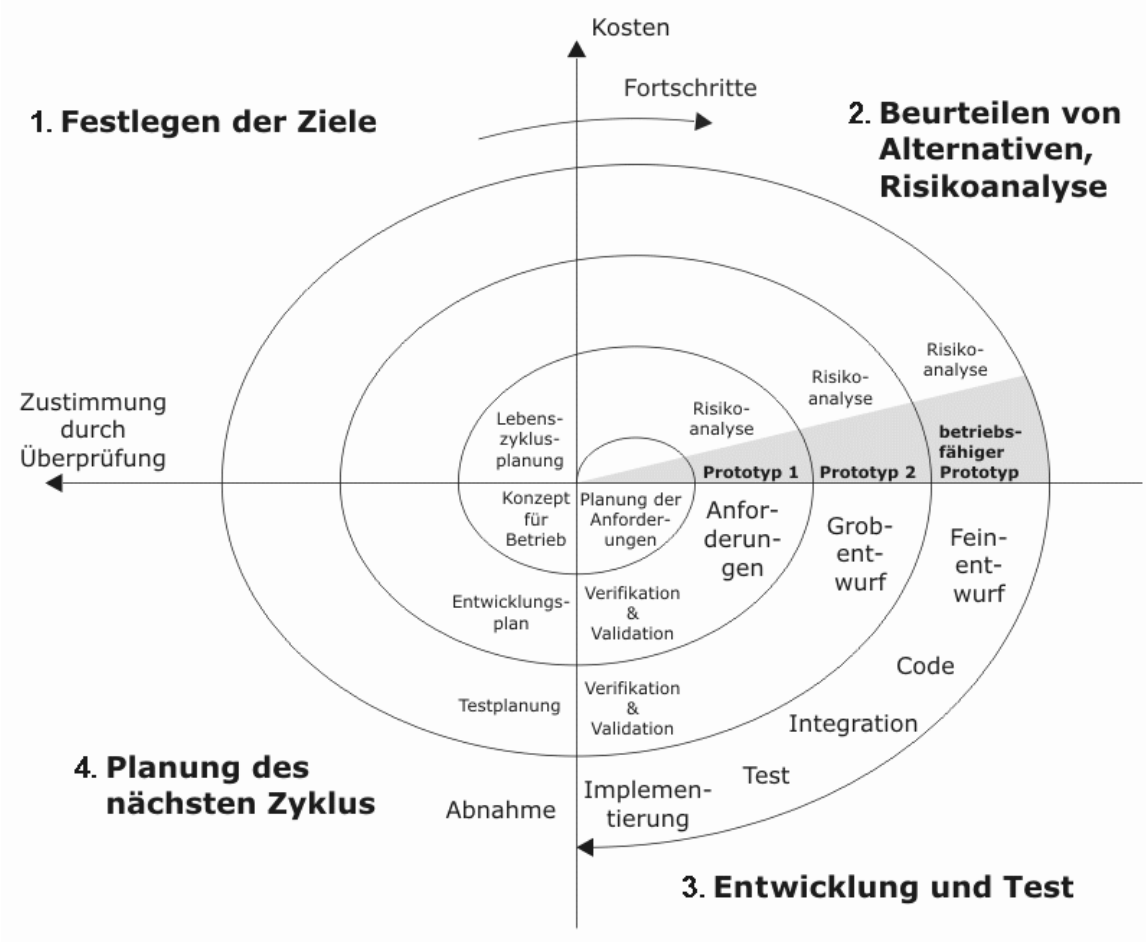
Hausaufgabenblatt 2

Patrick Gustav Blaneck

Letzte Änderung: 16. Oktober 2021

1. **Boehm'sches Spiralmodell**
   In der Vorlesung wurde ein wichtiger Meilenstein bei der Entwicklung von Software-Entwicklungsprozessen ausgelassen, das so genannte Boehmsche Spiralmodell. Informieren Sie sich über dieses Modell und beschreiben Sie die Hauptideen des Ansatzes. Überlegen Sie, in wie weit es sich um ein iteratives Modell und dann um ein inkrementelles Modell handelt. Dokumentieren Sie ihre Nachforschungen und Überlegungen.

Lösung:



Von dev-insider:

> Der Entwicklungsprozess durchläuft die Spirale von innen nach außen. Jede Umrundung stellt aufeinander aufbauende Phasen der Entwicklung dar. Ein Entwicklungsprozess beginnt im Inneren der Spirale mit Risikoanalyse, grundlegender Konzeptentwicklung und der Bestimmung von Anforderungen.
>
> Der erste Quadrant bezieht sich jeweils auf die Ermittlung von Zielen, der zweite auf die Bewertung von Alternativen und die Reduktion von Risiken. Im dritten Quadranten erfolgt die eigentliche Entwicklung, im Softwarebereich speziell die Implementierung. Der vierte Quadrant leitet schließlich mit der Planung der nächsten Phasen wieder in den ersten über.
>
> Als Risiko-orientiertes Entwicklungsmodell bezieht das Spiralmodell die Risiken der Prozessabläufe wesentlich in die Ausgestaltung der Struktur ein. Sie bestimmen sowohl den Grad der Detaillierung jeder einzelnen Stufe als auch, mit welchem Aufwand und Einsatz die jeweilige Stufe umzusetzen ist. Eine Risikobewertung ist daher in jedem Umlauf im zweiten Quadranten vorgesehen.
>
> Allgemein fokussiert das Spiralmodell auf die langfristigen Einflüsse über den gesamten Lebenszyklus eines Produkts. In Bezug auf Software bedeutet das vor allem ein Abrücken von einer zu starken Konzentration auf das Kodieren von Quelltext.

Aus Wikipedia:

> Das Spiralmodell gehört zu den inkrementellen oder iterativen Vorgehensmodellen. Es ist eine Weiterentwicklung des Wasserfallmodells, in der die Phasen mehrfach spiralförmig durchlaufen werden.
>
> Das inkrementelle und iterative Vorgehensmodell sieht daher eine zyklische Wiederholung der einzelnen Phasen vor. Dabei nähert sich das Projekt langsam den Zielen an, auch wenn sich die Ziele während des Projektfortschrittes verändern. Durch das Spiralmodell wird nach Boehm das Risiko eines Scheiterns bei großen Softwareprojekten entscheidend verringert.

Meine Nachforschungen und Überlegungen sahen wie folgt aus:

1. „Boehm'sches Spiralmodell“ googlen,
2. den Wikipedia-Artikel öffnen
3. um ihn dann wieder zu schließen,
4. das zweite Google-Ergebnis einfach überspringen,
5. im dritten Ergebnis eine fantastische Zusammenfassung finden und diese hier teilen!
6. Bemerken, dass ich noch betrachten muss, in wie weit es sich um ein inkrementelles bzw. iteratives Modell handelt,
7. den Wikipedia-Artikel erneut öffnen
8. und die wunderbare Erläuterung hier hinzufügen!

2. **Veränderte Anforderungen**

Einem Kunden fällt nach Zweidritteln erfolgreicher Projektlaufzeit ein, dass er eine bereits entwickelte Funktionalität nicht benötigt, dafür aber eine andere wünscht. Beschreiben Sie kurz, wie man bei Nutzung der folgenden Vorgehensmodelle darauf reagieren würde:

(a) Wasserfallmodell

Lösung:

Und Erklärung:

Wenn man erst einmal davon absieht, dass das Wasserfall-Modell nach Royce bewiesenermaßen ohnehin nicht funktioniert, sieht das Wasserfallmodell keine Möglichkeit der Reevaluation vor. Ergo wird bei striktem Ablauf nach dem Wasserfallmodell die Anforderung einfach ignoriert.

(b) Iterative Entwicklung

Lösung:

Man kann hoffen, dass die zu verändernde Funktionalität nicht zu viele Änderungen am bisher entwickelten System hervorruft. Ist dies der Fall, kann man in der nächsten Iteration die benötigten Änderungen vornehmen.

Sind allerdings zu viele Änderungen nötig, muss potentiell die gesamte Codebase verändert oder neugebaut werden. Das kann entweder einen (quasi) Neustart, oder das gleiche Ergebnis wie in (a) hervorrufen.

(c) Inkrementelle Entwicklung

Lösung:

Man kann hoffen, dass das zu verändernde Inkrement nicht zu viele Änderungen am bisher entwickelten System hervorruft. Ist dies der Fall, kann man in dem nächsten Inkrement die benötigten Änderungen vornehmen.

Sind allerdings zu viele Änderungen nötig, muss potentiell die gesamte Codebase verändert oder neugebaut werden. Das kann entweder einen (quasi) Neustart, oder das gleiche Ergebnis wie in (a) hervorrufen.

(d) Wo würden Sie das SCRUM Modell einordnen?

Lösung:

Aus Wikipedia:

> Der Ansatz von Scrum ist empirisch, inkrementell und iterativ. Er beruht auf der Erfahrung, dass viele Entwicklungsprojekte zu komplex sind, um in einen vollumfänglichen Plan gefasst werden zu können. Ein wesentlicher Teil der Anforderungen und der Lösungsansätze ist zu Beginn unklar. Diese Unklarheit lässt sich beseitigen, indem Zwischenergebnisse geschaffen werden. Anhand dieser Zwischenergebnisse lassen sich die fehlenden Anforderungen und Lösungstechniken effizienter finden als durch eine abstrakte Klärungsphase. In Scrum wird neben dem Produkt auch die Planung iterativ und inkrementell entwickelt. Der langfristige Plan (das Product Backlog) wird kontinuierlich verfeinert und verbessert. Der Detailplan (das Sprint Backlog) wird nur für den jeweils nächsten Zyklus (den Sprint) erstellt. Damit wird die Projektplanung auf das Wesentliche fokussiert.

3. **SCRUM**

Suchen Sie im Netz nach dem offiziellen Scrum-Guide von Ken Schwaber und Jeff Sutherland. Lesen Sie die aktuelle Version (englische Variante) und beantworten Sie anschließend die folgenden Fragen. (Hier zu finden: Scrum Guides)

(a) Welche Personen/Rollen sieht das Scrum-Modell vor?

Lösung:

- n Developers
- 1 Product Owner
- 1 Scrum Master

(b) Welcher Zeitraum ist laut Dokumentation vorgesehen für die Planung eines Sprints?

Lösung:

Aus dem Scrum Guide:

> Sprint Planning is timeboxed to a maximum of eight hours for a one-month Sprint. For shorter Sprints, the event is usually shorter.

(c) Welcher Zeitraum ist laut Dokumentation vorgesehen für die folgenden Abläufe / Meetings:

i. Daily Scrum

Lösung:

Aus dem Scrum Guide:

> The Daily Scrum is a 15-minute event for the Developers of the Scrum Team. To reduce complexity, it is held at the same time and place every working day of the Sprint. If the Product Owner or Scrum Master are actively working on items in the Sprint Backlog, they participate as Developers.

ii. Sprint

Lösung:

Aus dem Scrum Guide:

> They are fixed length events of one month or less to create consistency. A new Sprint starts immediately after the conclusion of the previous Sprint.

iii. Sprint Retrospektive

Lösung:

Aus dem Scrum Guide:

> The Sprint Retrospective concludes the Sprint. It is timeboxed to a maximum of three hours for a one-month Sprint. For shorter Sprints, the event is usually shorter.

iv. Sprint Review

Lösung:

Aus dem Scrum Guide:

> The Sprint Review is the second to last event of the Sprint and is timeboxed to a maximum of four hours for a one-month Sprint. For shorter Sprints, the event is usually shorter.

(d) Welche Nachteile und welche möglichen Probleme bringt das Vorgehen nach Scrum mit sich?

Lösung:

Reddit-User MoritzK_PSM fasst es sehr gut zusammen:

> Scrum makes certain implicit assumption about the people involved, that are not always given. A few examples:
>
> - A Scrum Team must be cross-functional and not have any sub-teams. That means people must usually go towards broadening the spectrum of their knowledge and ability. We speak of things like T-shaped and E-shaped skill profiles. Not everybody is *willing to do that* and in many places you still see, e.g., there being exclusively one person in a team for e.g. testing. This can create problems with delivery reliability, which may be better handled with a Kanban approach.
> - The whole idea of self-management is predicated on the idea that people *WANT to manage their own stuff*. Often enough you will find people that really don't want to take that responsibility and prefer to just follow instructions they are given by a manager.
>
>   An especially visible place is the Retrospective. If people do not care to improve their processes, it won't happen. A Scrum Master can only do so much in terms of encouraging and motivating. But people *who really don't care, won't care*.

- Scrum, in it's introduction, creates an initial disruption. It is meant to be one, to - as Scrum.org puts it - show the inefficacies of the existing processes. Now a lot of people don't like that, especially in the management. Here you can often see ScrumButs, e.g. „we introduce Scrum, but we don't have time for a Retro every Sprint" or „we do Scrum now, but all the QA is still handled by an external body", etc. Often a ScrumBut is still better than no Scrum at all, but also there are many cases where one aspect of Scrum missing drags other aspects down and the overall „faith" in the efficacy of Scrum is lost. Here a gradual approach might be better, gaining gradually more buy-in from the management.

Ultimately, Scrum does not solve your problems. Scrum is based on collective intelligence and pro-active engagement in the development. If you have shitty people, you won't get far with Scrum.