Softwareengineering

by Emad Easa

Wiederholung

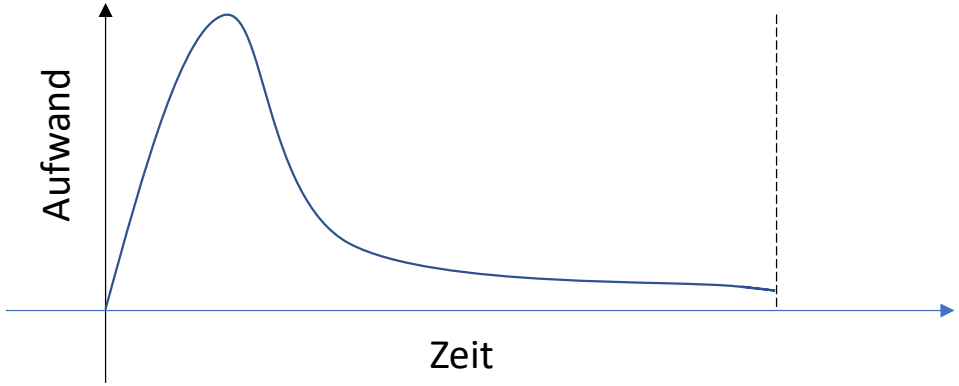
- Vorgehensmodelle definieren ein systematisches Vorgehen für die Umsetzung eines Projektes
- Die gängigsten Vorgehensmodelle lauten Build-and-Fix-Cycle, Software-Life-Cycle, Wasserfallmodell, V-Modell, Spiralmodell, Inkrementelles Modell, Iteratives Modell, Agiles Modell
- Die heutzutage gängigste Methode mit der ein Softwareentwicklungsprozess realisiert wird ist das Agile Modell

Projektmanagment - Aufwandsschätzung

- Wesentlicher Teil bei der Planung der Umsetzung eines Softwareprojektes
- Allgemeine Frage, die sich bei der Aufwandsschätzung ergibt: Sollte man die Zeit, die man benötigt eher über- oder sogar unterschätzen?
- → Hofstadter´s Gesetz: „Die Zeit, die man benötigt, um eine komplexe Aufgabe zu lösen wird immer mehr sein, als man schätzt, selbst wenn man das Hofstadtsche Gesetz mit einbezieht.“
- Ist die Lösung dann immer (viel) mehr zu schätzen?

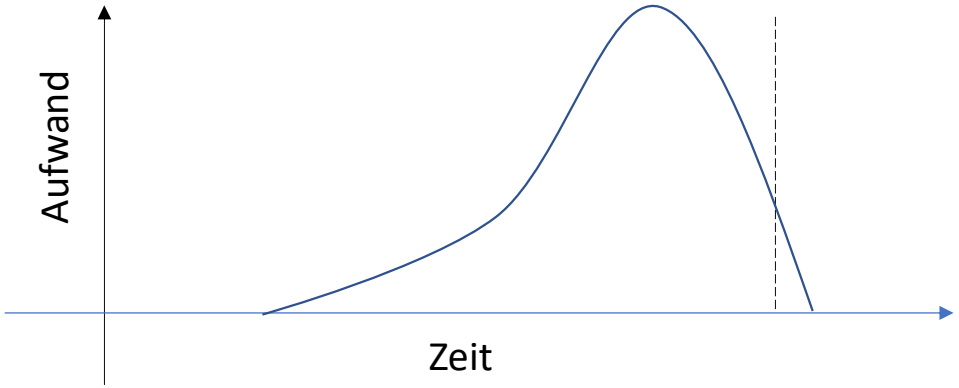
Projektmanagment - Aufwandsschätzung

- Parkinson´s Gesetz: Die Arbeit „verkompliziert“ oder „vergrößert“ sich immer, sodass die gesamt verfügbare Zeit zur Lösung einer Aufgabe in Anspruch genommen werden muss



Projektmanagment - Aufwandsschätzung

- Es gibt ein weiteres Phänomen, dass auftritt, wenn man „zu viel“ Zeit hat: Student´s Syndrom
- Das Student´s Syndrom besagt, dass man die Arbeit immer „in letzter Sekunde“ erledigt, falls man zu viel Zeit hat



Projektmanagment - Aufwandsschätzung

- Lösungen für Parkinson´s Gesetz und Student´s Syndrom?
- Kein Rezept aber folgende Praktiken helfen:
  - Team „Ermächtigung“ und „Puffer Planen“ (Implizite oder auftauchende Zeitpuffer an richtigen Platz im Projekt versetzen) um das Risiko des in die Länge ziehen von Arbeit zu minimieren
  - Agile Methoden wie Daily Standups, Sprint Retrospektiven und Reviews helfen den Fokus nicht zu verlieren

Team Ermächtigung

- Ein nicht ermächtigtes Team ist ein Team, dem alles egal ist
- Team Ermächtigung beschäftigt sich mit Fragen, ob das Team die Möglichkeiten besitzt über den Vorgang im Projekt zu entscheiden. Wenn dem nicht so ist, dann muss man dafür Lösungen finden.
- Z.B.:
  - Wurden Ziele definiert, die nicht erreichbar sind? (Kann zu Demotivation führen)
  - Kann das Team seine Stories im Sprint selbst definieren?
  - Entscheidet eine Teamfremde Personen über die Rollen im Team?
  - Werden Teammitglieder in ihrer Arbeit oft unterbrochen oder haben zu viele Verantwortungen?
  - ……

Gemeinsames Beispiel – Schätzen Zeitaufwand

- Zeit für ein kleines Experiment – jeder schreibt auf ein Stück Papier, wie lange es dauern würde, das Programm „Hunt The Knight“ in ein normales Schachprogramm umzuwandeln (Bedenke dabei, dass wir nicht von vorne mit der Implementierung beginnen müssen, da wir ja schon ein Basisprogramm haben. Bedenke weiters, dass wir bei allen Figuren implementieren müssen, wie sie ziehen und dass dabei auch beachtet werden muss, ob andere Figuren im Weg stehen) – Dokumentation nicht beachten
- **Schätze den Aufwand, den wir benötigen würden, um unser Monopoly Spiel fertig zu stellen (Bedenke, was noch fehlt: Karten, Fertigstellung des Spielfelds, Erstellung einer Spieldatenbank zur Spielstandspeicherung, Erweiterung in eine verteilte Anwendung, …) – Doku nicht beachten**
- Schätze den Aufwand für ein Schachprogramm / Monopoly mit grafischer Oberflächen

Vorgehensweise für die Projektschätzung

1. Definiere deinen Umfang:
   - Definiere deine User Stories und Features
   - Kalkuliere in deinem Umfang auch „nicht Software“ Ressourcen ein: Support Aufgaben, Meetings, Workshops, …. (jede Tätigkeit, die mehr als 5 Minuten in Anspruch nehmen kann)
   - Lass dein Team bei der Definition des Umfangs teilhaben

2. Schätze den Aufwand
   - Bei einem großen Projekt wäre es nicht möglich zu schätzen, ob ein Entwickler für eine Aufgabe 3 oder 6 Stunden benötigt
   - Dabei würde auch nur die zeitliche Komponente erfasst – das führt zur Gefahr der „Temporalen Illusion“ – jeder erfährt und schätzt Zeit anders
   - Besser sind relative Schätzungen, in bestimmten Größen, zum Beispiel nach Aufwand und das mit agilen Größen, wie zum Beispiel **Story Points**
   - Eine ideale Möglichkeit um Story Points zu vergeben bildet die sogenannte **Fibonacci Reihe** oder sogar eine **pseudo Fibonacci Reihe** (1, 2, 3, 5, 8, 13, 20, 40, 100, …) → Warum???

Weber´s Gesetz

- Annahme 1: Man hält eine Hantel mit einem kg in einer Hand und eine mit zwei kg in der anderen. Diese könnte man wahrscheinlich gut voneinander unterscheiden (Differenz von 100%)
- Annahme 2: Man hält eine Hantel mit 20 kg in einer Hand und eine mit 21 kg in der anderen. Diese könnte man nicht mehr unterscheiden (Unterschied 5%)
- Weber´s Gesetz beschreibt den „gerade fühlbaren Unterschied“. Auf Softwareprojekte bezogen bedeutet das, dass man nicht sagen kann ob eine Aufgabe eher 20 oder 25 Stunden benötigt, aber sehr wohl ob eher 20 oder 40
- Die Fibonaacci Reihe funktioniert gut, weil sie in etwa Weber´s Gesetz entspricht
- Ein anderer Ansatz wäre zum Beispiel die möglichen User Story Points immer zu verdoppeln: 1, 2, 4, 8, 16, 32, … → Gefahr hierbei wären vielleicht zu viele Diskussion ob ein Task wirklich doppelt so komplex ist wie ein anderer

„Playing Poker“ für die Projektschätzung

- Schätzen der Fibonacci Zahlen mit Pokerdeck
- Jeder im Team bekommt 4 Decks mit den Karten 1, 2, 3, 5, 8, 13, 20, 40, 100
- Für jede Aufgabe muss jeder im Team eine Karte abgeben.

„Playing Poker“ Resultat - Beispiel

Eine Runde „Playing Poker“ könnte z.B. folgendermaßen aussehen (die finalen Storypoints müssen kein Durchschnitt sein. Sie werden vom Team noch einmal gemeinsam besprochen und dann bestimmt)

User Story/Member	Pers1	Pers2	Pers3	Final SP
Story1	2	3	3	2
Story2	3	3	5	3
Story3	13	20	8	13
Story4	1	8	3	5
Gesamt				23

Die hier evaluierten Werte repräsentieren einen „optimistischen Aufwand“. Werte, die durch Erfahrungen und Meinungen der Teammitglieder verfälscht sein können. Ein besserer Ansatz wäre einen **Erwartungswert** zu ermitteln.

Projektschätzung – Erwartungswert Ermittlung

- Um den Erwartungswert für die Storypoints zu ermitteln wird oft die Methode „Program Evaluation and Review Technique” (PERT) verwendet.
- PERT verwendet folgende Gleichung:

$$s_e = \frac{a + 4m + b}{6}$$

- s_e = Erwartungswert, a = Best case, m = wahrscheinlichster Wert, b = worst case

Projektschätzung – Expected Value Ermittlung

- Nehmen wir an die Storypoints wären best case 20, voraussichtlicher Wert 13 und worst case – dann würde die PERT Gleichung folgendes ergeben:

$$s_e = \frac{8 + (4x13) + 20}{6} = 13.33 \text{ SP}$$

Diese PERT Gleichung ist keine spezifische für Softwareprojekte – sie beachtet nicht die für Softwareprojekte spezifischen Variablen und Probleme → Besser geeignet ist die **Stutzke Gleichung (pessimistisches PERT)**

Stutzke Gleichung

$$s_e = \frac{a + 3m + 2b}{6}$$

Auf das Beispiel angewandt ergibt das

$$s_e = \frac{8 + (3x13) + (2x20)}{6} = 14.5 \text{ SP}$$

Da keine Schätzung perfekt ist, muss man immer einen Puffer mit einbeziehen. Eine der einfachsten Methoden ist der sogenannte “Root Square Error (RSEM)”

$$\text{Puffer} = \sqrt[2]{\sum_{i=0}^{n} (a_i - bi)^2}$$

a = best case, b = worst case, n = Anzahl der features

Stutzke Gleichung

User Story/Member	Best Case	Worst Case	$(a_i - bi)^2$
Story1	2	3	1
Story2	3	5	4
Story3	13	20	49
Story4	5	8	9
Gesamt	23	36	
Buffer			7.9

Projektschätzung - Ergebnis

- Oft wird zu dem Puffer noch ein sog. „Contingency Buffer“ hinzugefügt, da man annimmt, dass der Puffer noch etwas zu gering ausfallen kann
- Es gibt verschiedene Methoden diesen „Buffer“ zu berechnen. Die einfachste davon ist etwa 10% noch hinzuzufügen:

  23 + 7.9 = 31.9 ~ 32 * 1,1 = 35,2 SP → Aufwand + Puffer + 10% Contingency = Gesamtaufwand

Gemeinsames Beispiel - Schätzen

- Aus dem Beispiel aus Folie 8 ersetzen wir die geschätzten Zahlen mit Zahlen aus der Fibonacci Reihe wobei zwischen 2 Fibonacci Zahlen immer die größere genommen werden muss. Verwende dabei einen Poker Dealer

Vorgehensweise für die Projektschätzung

1. Definiere deinen Umfang
2. Schätze den Aufwand
3. Schätzung die Zeit (Projektgeschwindigkeit):
   - Die Projektgeschwindigkeit ist die Zeit, die das Projektteam benötigt, um den Aufwand umzusetzen = die Zeit, die benötigt wird, um alle Aufgaben zu erledigen
   - Wenn wir einen Sprint für die zu erledigen Aufgaben definieren, dann ist die Projektgeschwindigkeit die Zeit, die benötigt wird, um alle Aufgaben eines Sprints zu erledigen
   - Dafür kann man folgende Methoden verwenden

Schätzung der Zeit

- Bester Fall: Mit jedem vom Team definieren, ob die definierten Tasks in den definierten Sprints möglich sind. Daraus resultiert, wie viele Storypoints / Sprints abgearbeitet werden können. Nehmen wir an es wären 10 SP / Sprint
- Wahrscheinlichster Fall: Entwickler haben in etwa eine Fehlerrate von 30%, wenn es um die Zeitschätzung geht. Dann sieht es mit den Storypoints / Sprint folgendermaßen aus:

  m = a * Effektivität → m = 10*0,7 = 7 SP / Sprint
- Schlimmster Fall: Im schlimmsten Fall können wir davon ausgehen, dass nur 10% aller Aufgaben in einem Sprint abgearbeitet werden.

  b = a * Erfolgsrate = 10 * 0,1 = 1 SP / Sprint

Schätzung der Zeit

- Nach der Stutzke Gleichung ergibt das folgendes:

$$v_e = \frac{10 + (3*7) + (2*1)}{6} = 5.5 \text{ SP / Sprint}$$

- Bei etwa 31 Storypoints ergibt das 31/5.5 = 5,64 Sprints → Wenn ein Sprint 2 Wochen dauert, dann ergibt das: 12 Wochen (aufgerundet)

Vorgehensweise für die Projektschätzung

1. Definiere deinen Umfang
2. Schätze den Aufwand
3. Schätzung die Zeit (Projektgeschwindigkeit)
4. Schätzung der Kosten
   - Die Projektkosten werden berechnet aus der Projektzeit x Durchschnittlichen Kosten einer Zeitperiode:

     $C_{Gesamt} = t * C_t$

Schätzung der Kosten

- Die Kosten des Projektes tragen sich aus folgenden Komponenten zusammen:
  - Kosten für die Entwicklung C_D
  - Kosten für die Infrastruktur C_I
  - Kosten für Transport und Reisen C_Z
  - Kosten für Support C_S
  - Sonstige Kosten C_O
- → $C_{total} = C_D + C_I + C_Z + C_S + C_O$

Schätzung der Kosten - Entwicklungskosten

- C_D = Anzahl Monate * $\sum_{i=0}^{n}(Gehalt_i + Steuerni) + B_D$

  B_D bildet hier in einen bestimmten Puffer.
- Was beinhalten Entwicklungskosten?
- Entwickler, Designer, Architekten, Tester, Manager
- Welche sind die Kostenrisiken, die man berücksichtigen muss?
- Unerwartbares Fehlen von Teammitgliedern, Tod, Projektgeschwindigkeit, …
- Lösung: Zum Beispiel maximales zusätzliches Gehalt für Unterstützung als Puffer mit einberechnen

Schätzung der Kosten - Infrastrukturkosten

- C_I = Anzahl Monate * $\bar{c}$ + B_i
- $\bar{c}$ sind die durchschnittlichen Kosten pro Monat
- Welche Kosten sind hier inkludiert?
- IT-Infrastruktur: Server, Lizenzen, Cloud Dienste, Elektrizität…
- Internet: Providerkosten, Hardwarekosten (Router, Switches, …)
- Büro: Miete, Sicherheit, Tische, …
- Equipment: Bildschirme, Drucker, Computer, …
- Sonstiges: Essen, Getränke, ….

Schätzung der Kosten - Transportkosten

- C_Z = Reise * $\sum_{i=0}^{n} (\bar{Z}) + B_Z$
- $\bar{Z}$ = Durchschnittliche Reisekosten pro Mitarbeiter, B_Z ist ein typischer Reisepuffer

- Welche Kosten sind durch Reisekosten gedeckt?
- Flüge, öffentliche Verkehrsmittel, Benzin, Hotel, Reiseversicherung, …

Schätzung der Kosten – Support-/Beratungskosten

- C_S = (Beratungszeit * L_H) + (Beratungszeit * C_F) + B_S

- L_H = Rechtlicher Beratugspreis pro Stunde
- C_F = Finanzieller Beratugspreis pro Stunde
- B_S = Puffer

Profit

- Um wirtschaftliche Gewinne zu erzielen werden zumeist Gewinne von 20%-30% verrechnet.
- Gewinn vor Steuer (EBT-Earnings Before Taxes) = Einkommen – Kosten
- Projektpreis = Kosten + Gewinnrendite (vor Steuer) + Steuern (in Ö 20% Ust. auf IT-Dienstleistungen = Durchlaufposten)
- Gewinn nach Steuern (EAT-Earnings After Taxes) für Beispiel GmbH in Ö: EAT = EBT / 1,25 (25% Steuersatz)

- Rendite = $\frac{\text{Einkommen - Kosten}}{\text{Einkommen}}$

Gemeinsames Beispiel

- Schätze die Projektkosten für Monopoly mit grafischer Oberfläche, Hunt the Knight, Schach, …

Zusammenfassung

- In der Projektplanung ist bei der Aufwandsschätzung ein „je mehr desto besser“ kein guter Ansatz
- Bei der Aufwandsschätzung, sollte man das Projektteam mit einbeziehen
- Die Zeitschätzung hängt von der Komplexität der Aufgaben (evt. In Storypoints) ab.
- In den Projektkosten gibt es einige Faktoren zu beachten, die auch von Projekt zu Projekt stark variieren können.

Quellenangabe

- Hofstadter´s Gesetz - https://en.wikipedia.org/wiki/Hofstadter%27s_law – letzter Zugriff 09.04.2022 18:44 Uhr
- Student´s Syndrom - https://en.wikipedia.org/wiki/Student_syndrome - letzter Zugriff 09.04.2022 18:44 Uhr
- Lösungen Parkinson´s Gesetz - https://www.solutionsiq.com/resource/blog-post/a-cure-for-parkinsons/ - letzter Zugriff 09.04.2022 18:44 Uhr
- Warum Fibonacci funktioniert - https://www.mountaingoatsoftware.com/blog/why-the-fibonacci-sequence-works-well-for-estimating - letzter Zugriff 09.04.2022 18:44 Uhr
- Aufwandsschätzung - https://www.offerzen.com/blog/estimating-software-projects-time-and-cost-like-a-pro - letzter Zugriff 09.04.2022 18:44 Uhr