

Dies ist keine „Musterlösung“, sondern eine gute von vielen möglichen Lösungen. Kommentare, die nicht Teil der Lösung sind, sind kursiv gesetzt.

Aufgabe 5-1:

- a. Nicht-funktionale Anforderungen (z.B. Zeitverhalten der Anwendung oder Benutzbarkeit) und Bereichsbedingungen (z.B. Datenmodelle, Schnittstellen zu Altsystemen)
- b. Sie sind aufwändig und machen es schwer, sinnvolle Verbesserungen zu realisieren. *Denn beim Beobachten sieht man nur, wie es die Leute derzeit machen, nicht wie sie es vielleicht besser machen sollten.*
- c. Keine. Man wird in der Regel sogar mehrere gleichzeitig anwenden. In Aufgabe 5-2 z.B. Interview, Szenarien und Anwendungsfälle.

Aufgabe 5-2:

1. **Inkonsistenzen:** Anfangs gibt das Sekretariat, am Ende die Dozenten Veranstaltungen ein.

Was macht man als Softwaretechniker mit Inkonsistenzen: Sicherlich wird man letztlich beim Kunden oder bei den betroffenen Stakeholdern (hier also z.B. erneut beim Sekretariat oder beim Dozenten) nachfragen müssen. In vielen Fällen wird man aber vorher für sich selbst die möglichen Alternativen und deren Bedeutungen durchdenken, damit man die richtigen Fragen stellen kann. In obigen Fall ist es nämlich wahrscheinlich so, dass die Eingabe von Veranstaltungen eine Tätigkeit ist, die sich aus Einzeltätigkeiten zusammensetzt, die gemeinsam vom Sekretariat und Dozent erledigt werden (z.B. sucht das Sekretariat freie Räume und trägt diese ein und der Dozent schreibt die Beschreibung für die Vorlesung). Beim ersten Interview ist dies nicht klar geworden, weil die Stakeholder oft ihr eigenes Tun nicht gut genug reflektiert haben, um es mit exakten Begriffen und allen Details zu erzählen.

2. **Anforderungen nicht-funktional vs. funktional**

- funktional
 - Sekretariat gibt Prüfungsleistung ein
 - Sekretariat gibt Veranstaltung ein
 - Dozenten geben Veranstaltungen ein
 - Studierende sehen Prüfungsleistungen ein
 - Studierende buchen Veranstaltungen
 - Studierende drucken Scheine aus.
- nicht-funktional (jeweils mit Messskalen):
 - „Bedienung aber nicht viel anders sein als beim KVV“ (Messen der Einarbeitungszeit)
 - Besondere Zuverlässigkeit wegen „rechtlichen Folgen“ (Anzahl korrekt laufender Testfälle)

„Dringend“ (Fertigstellungsdauer) ist eine Anforderung an den Prozess, nicht an das Produkt.

3. **Bereichsbedingungen:**

Die Studien- und Prüfungsordnungen der einzelnen Studiengänge werden erwähnt.

Mit den rechtlichen Folgen beziehen sich die SekretärInnen wahrscheinlich auf Hochschulgesetze oder bestehende Rechtsurteile.

Für den Informatiker ist dies auch nicht nur aus Datenmodellierungssicht wichtig über solche Bedingungen nachzudenken (d.h. aus der Studienordnung werden Analyseklassen wie Studiengang, Prüfung, etc. extrahiert), sondern auch im Hinblick auf die Funktionen des Programms. Sicher wünschen sich die SekretärInnen, in Zukunft keine Scheine mehr drucken zu müssen, aber wie rechtlich verbindlich ist ein solcher selbstausgedruckter Schein z.B. bei einer Bewerbung? Der Informatiker bewegt sich mit seinem solchen System also nicht im luftleeren Raum.

4. **Stakeholder:** Studierende, Sekretariat, Dozenten, Prüfungsamt, Systemadministrator, Studien/Prüfungsordnungsgeber, Geschäftsführung als Auftraggeber, evtl. auch die Entwickler.

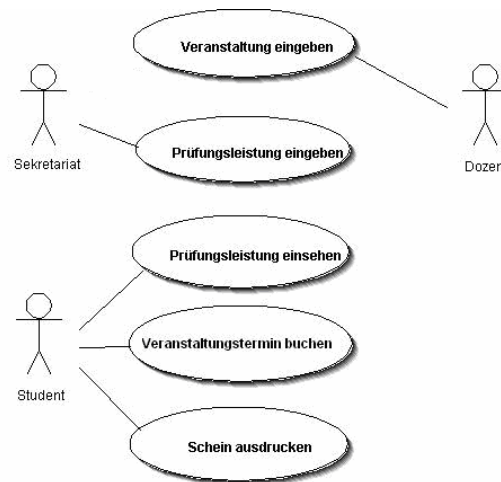
Beachte: Ein Stakeholder ist nicht unbedingt ein Akteur und nicht jeder Stakeholder ist Kunde. Der Kunde kauft die Software, der Akteur interagiert mit dem System, ein Stakeholder hat ein wie auch immer geartetes Interesse/oder gar nur eine Beziehung. In Aufgabenteil 1 haben viele von euch geschrieben „beim Kunden nachfragen“ und meinten damit die SekretärInnen.

Wer hier nur 4 nennt (wie ja verlangt) sollte vielleicht mit ... die Liste beenden, damit ein Korrektor auch das Gefühl bekommt, dass ihr zwar den Auftraggeber nicht benennt, ihr ihn trotzdem vielleicht wisst :-)

5.a. Akteure: Die genannten Akteure sind das Sekretariat, der Student und der Dozent. Man könnte auch den Drucker hinzunehmen. In der Regel sieht man von einem solchen Detail ab, aber man merke: Auch außen stehende, technische Geräte können Akteure sein, selbst solch passiven wie ein Drucker.

5.b. Anwendungsfälle: Siehe funktionale Anforderungen aus 2.

5.c. Anwendungsfalldiagramm:



Ein typischer Fehler: Eine Systemkomponente als Akteur darstellen, obwohl gerade das gesamte System als System unter Discussion diskutiert wird. Geht nicht, weil Akteure immer außerhalb des Systems sind, die Systemkomponenten aber jetzt auch Teil des betrachteten Systems ist.

Achtung: Wenn man für das „Veranstaltung eingeben“ Dozent und Sekretariat zugelassen hätte, wird es etwas schwieriger. Denn man kann nicht einfach beide Akteure assoziieren, denn das würde heißen, dass immer beide beteiligt sind. Es bleibt dann nur, einen generalisierten Akteur zu definieren (z.B. Eingabe) und Sekretariat und Dozent als Unterakteur (ähnlich wie Unterklasse) zuzulassen. Dieses Vererbungsbeispiel zeigt übrigens die große Flexibilität von UML.

Der Drucker würde dann zum Fall „Schein ausdrucken“ assoziiert. Natürlich fehlen einige wichtige Anwendungsfälle. Die wurden in dem Text aber nicht explizit erwähnt. Wenn Zweifel aufkommen, immer den Kunden fragen und nicht stillschweigend Annahmen treffen, die nachher nicht stimmen.

Glossar:

Sekretariat: Macht Verwaltungsaufgaben für eine Arbeitsgruppe

Dozent: Legt Veranstaltungen und deren Termine fest und führt Veranstaltungen durch

Student: Nimmt an Veranstaltungen teil und bekommt Prüfungsleistungen gutgeschrieben

Veranstaltung: Eine einsemestriger Kurs zur Erlangung von Prüfungsleistungen

Schein: Beleg einer Prüfungsleistung

Prüfungsleistung: Voraussetzung zur Erlangung eines Studienabschlusses des Studenten

6.a/b. Ereignisfolge:

Voraussetzung: Student ist angemeldet und hat eine Veranstaltung ausgewählt

1. Student wählt einen Termin (wenn es mehrere gibt, z.B. bei Übungen) für die Veranstaltung.
2. System stellt fest, dass der Termin noch einen freien Platz hat.
3. System stellt fest, dass die Veranstaltung für den Studenten belegbar ist.
4. System bucht den Termin und gibt positive Rückmeldung.

Zusicherung: Voraussetzung + Termin ist für Student gebucht + Termin hat einen freien Platz weniger

Man könnte die Voraussetzungen auch zu dem Anwendungsfall hinzunehmen, also dass sich der Student erst anmeldet und dann eine Veranstaltung auswählt. Beide sind aber auch selbst wieder Anwendungsfälle, die man dann idealerweise im Anwendungsfalldiagramm mittels <<include>> zu diesem in Beziehung setzt.

6.c. Erweiterungen

- 2a. Der Termin ist schon komplett voll. Die Buchung wird abgelehnt.
- 3a. Die Veranstaltung passt nicht zum Studienverlauf des Studenten. Die Buchung wird abgelehnt.
- 3b. Der Student ist schon zweimal durchgefallen (nur Vordiplom). Es wird eine Studienberatung empfohlen.
- 3c. Wenn der Student für das Semester zu viele Leistungspunkte bucht, kommt eine Warnung.

Achtung: Es ist selten sinnvoll „katastrophale“ Fehlerfälle zu berücksichtigen, wie z.B. „Internet stürzt ab“, „Datenbank nicht erreichbar“, da diese den Use-Case unnötig aufblähen und man die Behandlung dieser lieber global erledigen will (sie können ja auch in vielen anderen Use Cases auftreten).

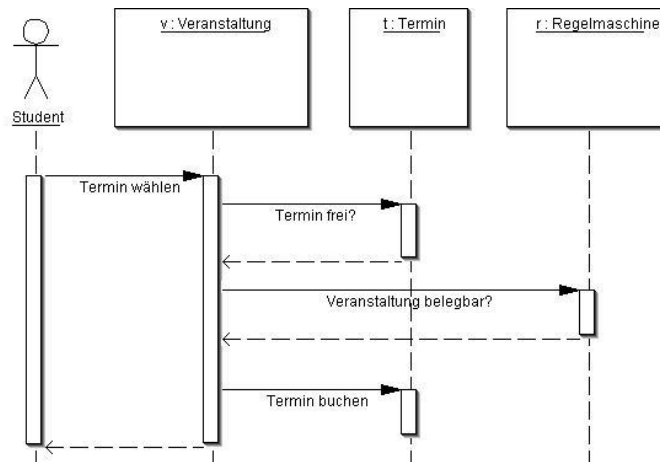
6.d.

Sequenzdiagramm oder Aktivitätsdiagramm. Ein Sequenzdiagramm ist gut geeignet, um den Haupterfolgsfall darzustellen (Erweiterungen sind nämlich gar nicht so einfach darzustellen, ihr

könnt es gerne mal versuchen). Gibt es viele Alternativen/Ausnahmen, dann ist ein Aktivitätsdiagramm wahrscheinlich sinnvoller.

Beim Sequenzdiagramm muss man sich aber die Klassen der Objekte überlegen und etwas weiter aufschlüsseln, was das „System“ eigentlich ist. Das ist aber schon Teil der Anforderungsanalyse. So gesehen sind Sequenzdiagramme ein gutes Mittel, systematisch von der Erhebung zur Analyse zu gelangen.

Das Sequenzdiagramm für den Erfolgsfall sieht z.B. so aus:



Die Beziehung der Objekte wird hier nicht geklärt (z.B. dass t ein Termin in v ist). Hier wurde eine Regelmaschine eingeführt, die die Studienordnungen kennt (und leicht änderbar ist, da sich Studienordnungen auch gelegentlich mal ändern). Achtung: Der Begriff ist schon sehr technisch und nimmt eigentlich unnötigerweise schon eine Entwurfsentscheidung voraus, die hier noch gar nicht gefragt ist.

Aufgabe 5-3:

Aufgabe ist es sich zu überlegen, welche Folgen diese Anforderungsänderung hat, wenn die Software schon fertig gestellt ist. Ideen: Die Software des Praxishelferrechners muss um diesen Fall erweitert werden, die Helfer brauchen entsprechende Autorisierungen, die softwareseitige Koordination bei gleichzeitigen Erfassungen muss erweitert werden (Entwurf), die Dokumentation und Schulungen müssen erweitert werden (und letztere evtl. verlängert), das System muss auf implizite Annahmen abgeklopft werden, dass die Verordnung immer von einem Arzt stammt und sicherlich vieles mehr, was wir uns als Systemfremde gar nicht denken können...

Aufgabe 5-4:

- Alle:
 - VO-1 bis 3 leiden unter Passivphrasen, damit ist für den Leser nicht gut zu erkennen, wer eine Tätigkeit initiiert und wer sie durchführt. In VO-1 ist hierdurch auch unklar, welche Rolle die Administrative Kraft in dem Anwendungsfall einnimmt.
 - Das System wird zu selten benannt (kein Ping-Pong, keine Ein-/Ausgaben). So fragt man sich z.B. bei VO-1, ob der in Schritt 1 ermittelte Kostenträger auch in das Informationssystem des Ordnungsgebers (taucht in Schritt 2 auf) eingetragen werden muss, oder ob letztlich hier nur der Kostenträger bestimmt wird, ohne dass etwas mit dem neuen Wissen geschieht.
- VO-1:
 - Feste Begrifflichkeiten werden zu locker verwendet: VO-1 Kurzbeschreibung: „Einzelleistungen werden verordnet“, Ablauf: „Erfassung einer Verordnung“.
 - Schritt 1 wird durch die Erklärung was es für Kostenträger geben kann unnötig überladen, ohne das hierdurch ein besseres Verständnis über den Ablauf erlangt wird. Wieso nicht einfach:
 - Der Ordnungsgeber bestimmt den Kostenträger der Verordnung. (mit einem Verweis auf den gesondert behandelten Fall)
- VO-2:
 - Die Kurzbeschreibungen weichen in ihrem Wortlaut deutlich von dem tatsächlich beschriebenen Verlauf ab, so wird in der Kurzbeschreibung von der Festlegung eines Typs gesprochen im Ablauf taucht dies aber nicht auf.
 - In 2. wird eine Erweiterung (wenn mehrere...) im Hauptablauf beschrieben, die viel zu technisch für das aktuelle Beschreibungsniveau ist. Dass Einzelpositionen angelegt werden gehört eher in einen Detail-Use Case,
- VO-3:
 - Use-Cases sind nicht wirklich geeignet solche fundamental-unterschiedlichen Abläufe wie elektronische vs. handschriftliche Signatur in einem Diagramm darzustellen. Es würde sicher keiner von euch auf die Idee kommen einen Anwendungsfall Flugbuchung im Internet zu nennen und im selben Fall die Buchung via Telefon als Alternative zu beschreiben.
 - In der Alternative liebt der „papiergebundene Prozessweg“ leider ein etwas unklares Konzept.