USE-CASE-Diagramme

- dt.: Anwandungsfalldiagramm

Problem:

- jede Software muss ganz bestimmte Anforderungen erfüllen
- Anforderungen werden vom Auftraggeber bestimmt
- je später neue Anforderungen im Enticklungsprozess bekannt werden desto teurer wird deren Entwicklung, da eventuell bereits bestehende Funktionalitäten überarbeitet werden müssen
- ideal wäre wenn am Anfang alle benötigten Anforderungen bekannt wären
  => Use-Case Diagramme

Definition:

- soll einen Überblick vermitteln, was die zu entwickelnde Software können muss
- wichtige Funktionen werden herausgearbeitet und zueinander in Verbindung gesetzt
- ohne Beachtung der technischen Implementierung

Aufbau:

- es gibt den Akteur und das System (Bild1)
- Akteur = Anwender
- System = zu entwickelnde Software (als Rechteck dargestellt)
- wesentliche Anforderungen werden im System platziert => knappe Funktionsbeschreibung in Ellipsen, eine Ellipse = eine Funktion
- Ellipsen sind die Use-Cases
- Zusammenhang zwischen Use-Cases und Akteur über Verbindungslinien mit Schlüsselwörtern dargestellt (Bild2)
- Verbindungslinien = Assoziationen
- 2 Arten von Verbindungslinien: durchgezogen und gestrichelt
- durchgezogen bedeutet, dass der Akteur den Use-Case anwendet (tauscht Informationen aus), z.B Start einer Funktion des Systems oder Funktion gibt Daten aus
- gestrichelt stellt Assoziation zwischen 2 Use-Cases dar
- 2 verschiedene Arten von Assoziatonen zwischen Use-Cases (include, extend)
- dargestellt mit <<include>> oder <<extend>> an den Linien
- include heißt, dass der Use-Case von dem die Verbindungslinie ausgeht den anderen einschließt (im Beispiel: B wird gestartet wenn A läuft und A wird fortgesetzt wenn B zuende ist)
- extend heißt, dass der Use-Case von dem die Verbindungslinie ausgeht den anderen möglicherweise erweitert, muss aber nicht (im Beispiel: D wird ausgeführt wenn bestimmte Bedingungen von C wahr sind)
- Bedingung von extend-Verbindungen müssen angegeben werden (Bild3)
- Use-Case wird um einen Extension Point (Erweiterungspunkt) erweitert, der in der extend-Bedingung angegeben wird damit man weiß, wann die Bedingung geprüft wird
- ein Use-Case kann beliebig viele extend-Verbindungen und somit auch Extension Points besitzen

Praxisbeispiel:

- Beispiel Online-Shop (Bild4)
- Strichmännchen ist der Kunde => Online-Shop aus Sicht des Kunden
- System bietet dem Kunden 5 Funktionen
- Assoziation zwischen Akteur (Kunde) und Use-Case nicht zwingend notwendig (Bild5)
- Kunde hat keinen Einfluss wann und ob Adresse überprüft wird
- include bedeutet, dass immer wenn eine Adresse angegeben wird diese überprüft wird
- Bild6 => extend bedeutet, dass Use-Case ausgeführt wird wenn Bedingung erfüllt ist (Bankeinzug gewählt)
- „Daten für Bankeinzug eingeben“ erweitert Zahlungsmethode wählen

- Bild7 => Online-Shop aus Sicht des Distributors (der, der die Bestellungen ausliefert)

Zusammenfassung:

- Use-Case-Diagramme sind Verhaltensdiagramme (beschreiben bestimmte Aspekte, wie sich ein System verhält)
- keine Möglichkeit die Reihenfolge der Ausführung festzulegen (nur bedingt über include/extend)
- z.B. im Bsp. nicht möglich Kunden- und Bestelldaten nur dann runterzuladen wenn Zugriffsberechtigung vorhanden sind
- extend macht keinen Sinn, da es dann in den Use-Case „Zugriffsberechtigung prüfen“ eingeschlossen wäre (hat grundsätzlich nichts miteinander zu tun)
- dafür sind sie aber auch nicht vorgesehen, dafür gibt es andere Diagramme

Quelle: http://www.highscore.de/uml/usecasediagramm.html