0.1 Objektorientierung

Grundidee: Reale Welt besteht aus Objekten, die untereinander in Beziehungen stehen.

- Klasse:
  - Daten(Attribute)
  - Funktionalität(Operationen, Methoden)
- Objekte:
  - In der Lage Nachrichten (= Methodenaufrufe) zu empfangen
  - Daten verarbeiten
  - Nachrichten senden
  - können einmal erstellt werden
  - in verschiedene Kontexten wiederverwendet werden

Objektorientierte Analyse(OOA): Objekte-Konzepte-in Domäne zu finden und zu beschreiben.

Objektorientierten Design (OOD): Geeignete Softwareobjekte und ihr Kollaboration zu definieren um Anforderungen zu erfüllen.

Domänenschicht: Klassen abgeleitet aus dem Domänenmodell (Low-Representational-Gap)

0.1.1 Use Cases und System-Sequenzdiagramm

Basis für das Design:

1. Szenarien
2. Systemoperationen
3. Domänenmodell

Was programmiert werden muss:

1. Systemoperationen bzw. deren Antworten

Use-Case-Realisierung: Wie ein bestimmter Use Case innerhalb Design mit kollaborierenden Objekten realisiert wird.

Systemoperationen: Jedes Szeanario schrittweise entworfen und implementiert

UML-Diagramme: Gemeinsame Sprache um Use-Case-Realisierungen zu veranschaulichen und zu diskutieren.

0.1.2 Klassen entwerfen:

Zwei arten von Design-Modellen (ergänzen sich und werden parallel erstellt):

- Statische Modelle:
  - UML-Klassendiagramm- Unterstützen Entwurf Paketen, Klassennamen, Atrributen und Methodensignaturen (ohne Methodenkörperd)
- Dynamische Modelle:
  - UML-Interaktionsdiagramme Untersützen Entwurf der Logik, des Verhaltens des Codes und der Methodenkörper.

0.2 UML-Diagramme für das Design

0.2.1 Grundelemente der UML

- Primitiver Datentyp
- Literal
- Schlüsselwort, Stereotyp
- Randbedingung (constraint)
- Kommentar
- Diagrammrahmen (optional)

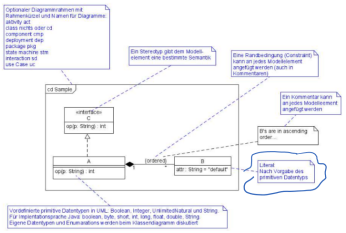

Abbildung 1: Grundelemente_Klassendiagramm.

0.3 UML-Klassendiagramm

- Statische struktur
- Konzeptuell: Problemdomäne (Domänenmodell)
- Design: Lösungsdomäne (DCD)

0.3.1 Notationselemente

- Klasse
- Attribut
- Operation
- Sichtbarkeit von Attributen und Operationen
- Assoziationsname, Rollen an den Assoziationsenden
- Multiplizität (Objekte der betreffenden Klasse)
- Navigierbarkeit in Assoziationen
- Datentypen und Enumerationen
- Generalisierung / Spezialisierung
- Abstrakte Klassen
- Komposition
- Aggregation
- Interface

- Interface - Realisierung (Menge von öffentlichen Operationen, Merkmalen und Verpflichtungen, die durch eine Klasse, die die Schnittstelle implementiert, zwingend zur Verfügung gestellt werden müssen.
- Assoziationsklasse (Da wenn ** Beziehung existiert)
- Aktive Klasse (Instanz wird in einem separaten Thread ausgeführt)

0.4 UML-Interaktionsdiagramme

Spezifiziert, auf welche Weise Nachrichten und Daten zwischen Interaktionspartnern ausgetauscht werden.

2 Arten:

1. Sequenzdiagramm
2. Kommunikationsdiagramm

Anwendung: Kollaborationen bzw. Informationsaustausch zwischen Objekten zu modellieren.

- Wer tauscht mit Wem Information aus?
- in welcher Reihenfolge
- Zeitlicher Ablauf
- Schachtelung und Flussteuerung (Bedingung, Schleifen, Verzweigungen) möglich.

Kann in mehreren Perspektiven verwendet werden:

- Analyse
  - mit SSD Input-/Output-Erignisse (Systemoperationen mit Rückgabeantworten) für ein Szenario eines Use Cases modelliert
- Design
  - mit SSD Interaktion zwischen Objekten zur Realisierung eines konkreten Use-Case Szenarios zu modellieren

Notationselemente:

- Lebenslinie
- Aktionssequenz
- Synchrone Nachricht
- Antwortnachricht
- Gefundene, verlorene Nachricht
- kombiniertes Fragment

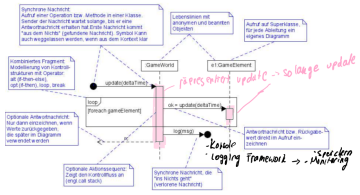

Abbildung 2: interaktionsdiagramm1