

Programmierpraktikum

Sommersemester 2007 - 29.06.2007

Björn Wilmsmann, B.A.
Sprachwissenschaftliches Institut
Ruhr-Universität Bochum
wilmsmann@linguistics.rub.de

Heute

- Spezifikation der zu entwickelnden Anwendung
- Formulierung des Lastenhefts (Planungsphase), wird in unserem Fall auch als Pflichtenheft dienen
- Modellierung anhand von Use Case, Klassen-, Sequenz- und Aktivitätsdiagrammen (Definitionsphase)

Lastenheft

- grobes Pflichtenheft
- dient als Grundlage für Machbarkeitsstudie und Kostenabschätzung
- Was soll die Applikation können?
- Welche Anforderungen soll die Applikation erfüllen?

Lastenheft

- Zielbestimmung
- Produkteinsatz
- Produktübersicht
- Produktfunktionen (z.B. notiert mit /LFnn/)
- Produktdaten (z.B. notiert mit /LDnn/)
- Produktleistungen (z.B. notiert mit /LLnn/)

Lastenheft

- Qualitätsanforderungen
- Ergänzungen

Zielbestimmung

- die Applikation soll
 - Webseiten anhand einer Liste einlesen und den HTML-Code parsen und in Klartext umwandeln
 - die Sprache der Webseite erkennen
 - die Webseite in vorher festgelegte Kategorien einordnen

Zielbestimmung

- die Inhalte sollen nach Kategorien sortiert abgespeichert werden
- es soll eine Suche implementiert werden, die eine Beschränkung der Suchergebnisse auf bestimmte Kategorien ermöglicht

Produkteinsatz

- zunächst die Teilnehmer des Kurses

Produktübersicht

- Umweltdiagramm

Produktfunktionen

- ...

Produktdaten

● ...

Produktleistung

● ...

Qualitätsanforderungen

- ...

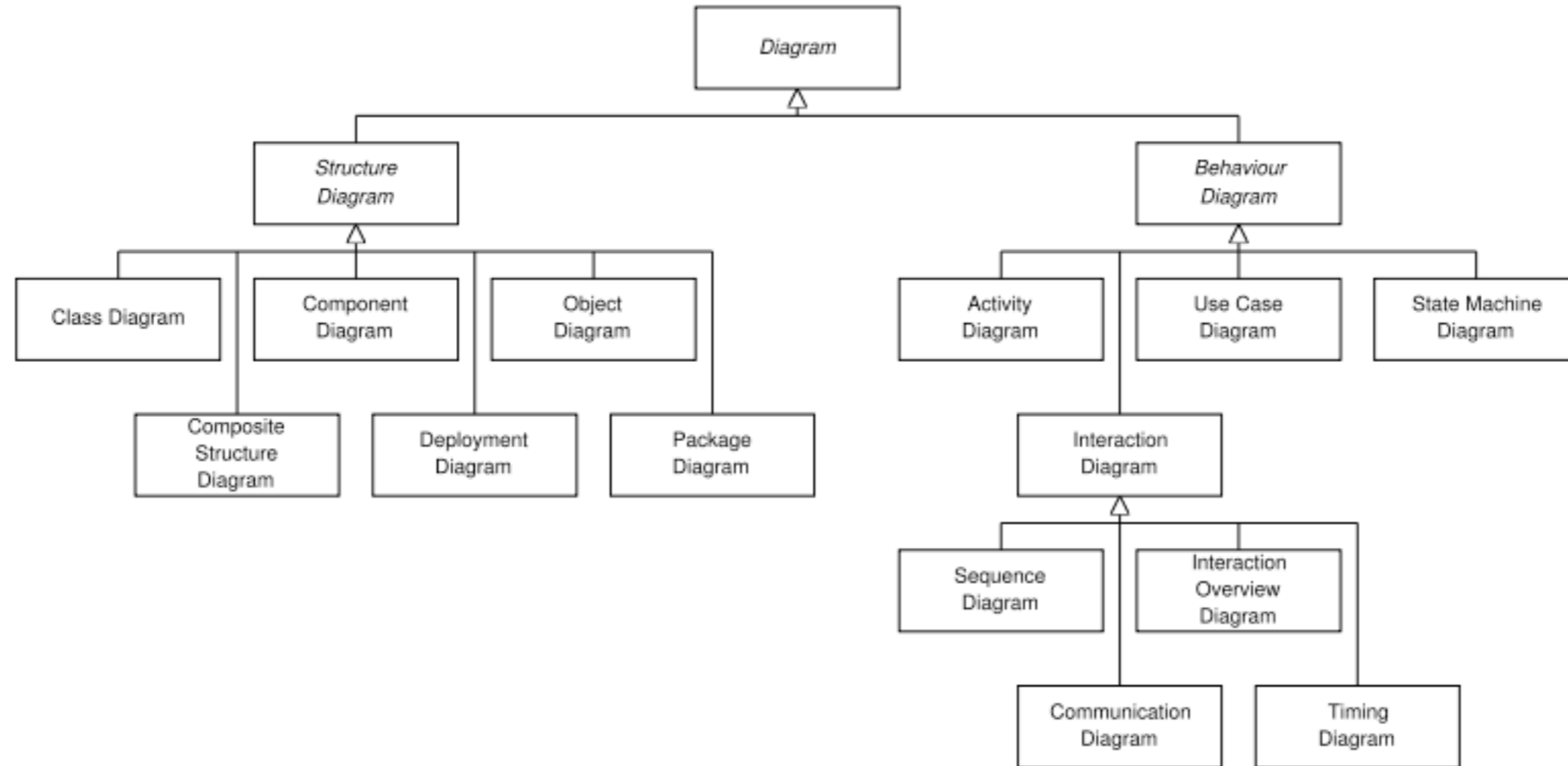
Ergänzungen

● ...

Modelle

- die unserem Fall verwendeten Modelltypen sind Teil der UML (Unified Modelling Language)
- von der OMG (Object Management Group) herausgegebene Spezifikation zur Modellierung von Softwaresystemen (bzw. seit Version 2.0 auch für Systeme im Allgemeinen)

Modelle



Modelle

- Diagramme ermöglichen jeweils unterschiedliche Sichten auf das zu entwickelnde Softwaresystem
- ermöglichen planvolles Vorgehen beim Programmieren
- im Idealfall ist die Software durch die Diagramme vollständig definiert

Modelle

- Erstellung idealerweise mit einem CASE (Computer-Aided Software Engineering) Tool (direkte Code-Generierung aus Modell!)
- Erstellung in unserem Fall mit Visual Paradigm (Alternativen: z.B. Rational Rose von IBM, EMF der Eclipse Foundation, EclipseUML von Omondo, ArgoUML)

Use Case Diagramme

- spezifiziert die Anwendungsfälle eines Systems
- Vorsicht: Häufige Verwechslung von Anwendungsfall (Interaktion der Umwelt mit dem System, daher auch Umweltdiagramm) und Geschäftsprozess (interne Abläufe eines Systems)

Klassendiagramme

- beschreibt in der objektorientierten Welt die Beziehungen von Klassen und Objekten untereinander
- Stichworte: Klasse, Objekt, Assoziation, Assoziative Klasse, Aggregation, Komposition, Vererbung, Abstrakte Klasse, Design Patterns

Sequenzdiagramme

- dient der Veranschaulichung von zeitlichen Abläufen zwischen Objekten
- Stichworte: Aktivität, Botschaft, Lebenslinie

Aktivitätsdiagramme

- dienen der Beschreibung algorithmischer Abläufe
- Variante von Zustandsautomaten bzw. des Zustandsdiagramms (state diagram)
- funktional vergleichbar mit den Flussdiagrammen, Struktogrammen nach Nassi, Shneiderman und Programmablaufplänen (PAP)

Referenzen

- [1] Balzert, Helmut (2000): Lehrbuch der Software-Technik. 2.Aufl. Heidelberg: Spektrum.

Referenzen

- [2] GoF (1995): Design Patterns - Elements of Resuable Object-Oriented Software, Boston, MA:Addison-Wesley.

Referenzen

- [3] <http://www.omg.org/>

**Vielen Dank für Ihre
Aufmerksamkeit!**