Veranstaltungsnummer: 18.219

Systementwicklung mit Objekten, Komponenten und Agenten

Ort: C-221
Zeit: Freitags 10–12

+++ NEU: Heiko Rölke: Übungen 12–14 +++

Daniel Moldt[1]
Fachbereich Informatik, Universität Hamburg,
Vogt-Kölln-Straße 30, D-22527 Hamburg,
Gebäude C, Raum C-220
Tel.:(+49) 40 - 428 83 22 47 / 36,
email: moldt@informatik.uni-hamburg.de

"Der Weg ist das Ziel."
"Der Weg entsteht beim Gehen."
Laotse

⇒ "Das Ziel entsteht beim Gehen."

There are more things in heaven and earth, Horatio,
than are dreamt of in your philosophy.

William Shakespeare, Hamlet

[1] Systementwicklung mit Objekten, Komponenten und Agenten Daniel Moldt ©2000

Was ist das Ziel?

- Spezifikationsproblematik erkennen
- Methode kennenlernen
- Darstellungstechniken kennenlernen
- Petrinetze vertiefen
- Kennenlernen eines ganzheitlichen Ansatzes zur Spezifikation und Systementwicklung

Was ist der Weg?

- Vorstellen der Objektorientierten Analyse
- Vorstellen der Komponentenorientierung
- Vorstellen von Petrinetz-Definitionen
- Vorstellen des Prototyping
- Vorstellen des Agentenkonzeptes
- Vorstellen eines netzbasierten Ansatzes

Was ist das Gehen?

- Üben anhand von kleinen / mittleren Beispielen
- Nutzung von Werkzeugen
- Anwendung der erworbenen Kenntnisse (in der Praxis)

Spezifikation von komplexen Systemen

- verschiedene Sichten
- Techniken, Methoden, Werkzeuge
- Bsp. Aufzug, CIM, Bank, Börsenspiel

Querbezüge zu aktuellen Themen

- Objektorientierung
- Komponenten
- Agenten
- Workflow, Geschäftsprozesse
- Prototyping
- Internet/Intranet
- VKI, Multiagentensysteme

Weitere Ziele:

- Kennenlernen von: Techniken, Vorgehensweisen, Werkzeugen
- Übung
- Ausbildung zum Systemanalytiker

Systementwicklung bzw. Spezifikation lassen sich nicht einfach beschreiben. Deshalb werden im Laufe der Vorlesung verschiedene Varianten vorgestellt. Dabei werden wir sehen, daß die Voraussetzungen von zentraler Bedeutung sind.

Im folgenden werden kurz zentrale Begriffe (Systemanalyse, Spezifikation, System) und dabei insbesondere Spezifikationsansätze (Strukturierte Analyse, Objektorientierte Analyse, Prototyping, Agenten, Workflow, Internet / Intranet) diskutiert.

Was ist Systemanalyse?

- Systemanalyse als Begriff, der eng mit dem Phasenmodell zusammenhängt.
- Was soll das System leisten?
- Abgrenzung zum Systementwurf:
  Nicht: Wie soll das System etwas leisten?
- Es gibt Methoden für die Systemanalyse.
- Es gibt Techniken für die Systemanalyse.
- Es gibt Werkzeuge für die Systemanalyse.

Wo liegt die Spezifikationsproblematik?

There are more things in heaven and earth, Horatio,
than are dreamt of in your philosophy.

William Shakespeare, Hamlet

- Die Anwendung wird unterschätzt.
- Die Arbeitsmittel werden nicht richtig verstanden und eingesetzt.
- Es fehlen die "richtigen" Methoden, Techniken, Werkzeuge.

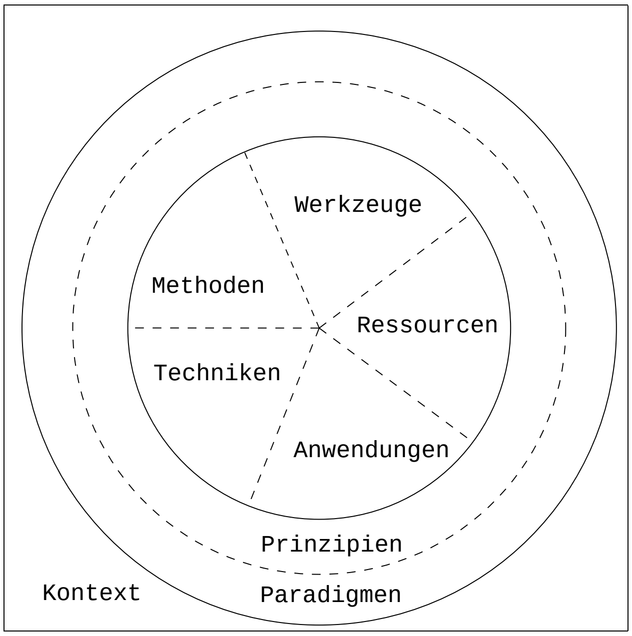
⇒ Das Spannungsfeld Methode – Techniken – Werkzeuge – Ressourcen – Anwendung wird nicht beherrscht.

Welche Rahmen für die Softwareentwicklung?

- Organisatorische Umgebung
- Hard- und Software Umgebung
- Brainware Umgebung
- Vorgehensmodelle
- Phasen?
- Modelle und Sichten
- Qualitätssicherung
- Konzepte und Strukturen
- Anwendungsspezifische Daten
- Soziologische Projektkomponenten
- Betrieb(swirtschaft)liche Projektkomponenten
- Existierende Lösungen
- etc.

Ansatzbegriff

Systematische Betrachtung für Aufgabenstellungen der Systemspezifikation
mit Beschränkung auf die technischen Aspekte der Systemspezifikation



1. Einführung

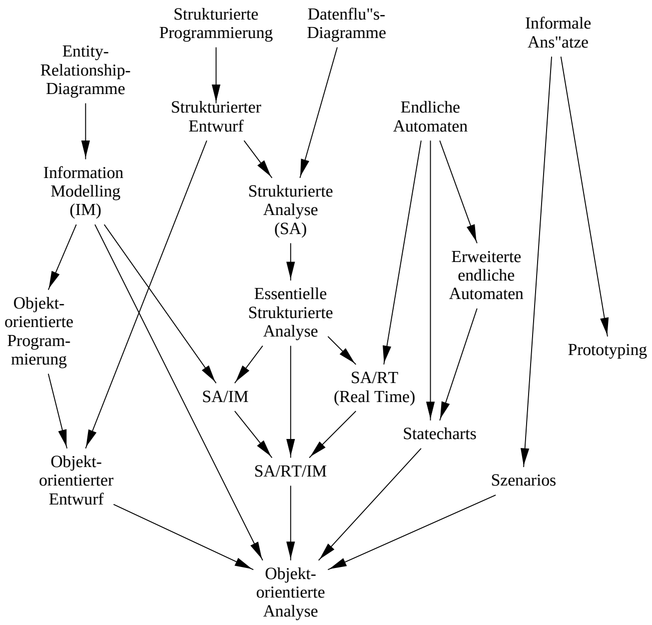
Vergleich von Entwicklungen

Spezifikation		Programmierung
Funktional dynamisch statisch objektorientiert	Paradigmen	Funktional imperativ objektorientiert logisch
ERD, DFD, EA, CFD, SC, KD, Netze	Technik	Fortran, C, Cobol, Lisp, ML, Smalltalk
SA, Statechart, IM, OOA	Methode	SP, OOP
Editor, Simulator, Animator, Analysator, Repository	Werkzeug	Editor, Compiler, Debugger, Programmierumgebung

- **Folgerungen:**
  - Die Entwicklungen im Bereich der Spezifikation verläuft wie in der Programmierung:
    kontinuierlich und meist evolutionär.
  - Verbesserungen einer der Facetten erlauben Verbesserungen der anderen Facetten.
  - Es fehlten bisher normierende, formale Semantiken für die Techniken, Methoden für formale Techniken und Werkzeuge für neueste Ansätz.

Allgemein gilt: Spezifikationsansätze werden formaler, benutzerfreundlicher, mächtiger und zunehmend durch Werkzeuge unterstützt.

Ansatzbegriff



../graphiken/Agenten.eps

Abbildung 1: Abhängigkeiten der untersuchten Ansätze

1. Übung:

Spezifikation des Aufzugs in Gebäude F
Beschreibung des Ist-Zustandes
In diesem Zusammenhang wichtige Fragen:

- Was sind Deine Probleme bei dieser Aufgabe?
- Wie versuchst Du diese Probleme zu beseitigen?
- Warum lassen sich nicht alle Probleme beseitigen?
- Welche Bedeutung haben Rahmenvorgaben für Deine Tätigkeit?
- Wie kannst Du die Aufgabenstellung für Dich (und andere Beteiligte) herausarbeiten?
- Was umfassen die Dir zur Verfügung gestellten Mittel (hier und sonst)?
- Setzt Du explizit und bewußt ein methodisches Vorgehen ein?
- Verwendest Du bewußt bestimmte (ausgewählte) Techniken, um deine Arbeit zu unterstützen, ermöglich, verbessern, optimieren, kommunizieren etc.?
- Welche Erfahrungen hast Du bzgl. der Spezifikation von Systemen?
- Bist Du mit den Ergebnissen Deiner Spezifikation zufrieden?
- Wer kann die Ergebnisse wie verwenden?
- Welche Vorteile siehst Du in einer Spezifikation?
- Welchen Unterschied siehst Du in bezug auf Entwurf und Implementation?

Wichtige Fragen zur Übung:

- Berücksichtigt bei den folgenden Fragen und Antworten die Arbeit vor und nach dem Zusammenschluß.
- Welche Techniken habt Ihr eingesetzt?
- Welche Methode habt Ihr eingesetzt?
- Welche Modelle habt Ihr erstellt?
- Wie weit seit Ihr gekommen?
- Wie gut versteht Ihr die Anwendung?
- Warum habt Ihr diese Spezifikation gewählt?
- War Euer Vorgehen richtig?
- War die Aufgabenstellung ausreichend?
- Wie hoch schätzt Ihr den weiteren Zeitaufwand bis zum ersten "vernünftigen" Modell?
- Welche Probleme hattet Ihr mit Euren Nachbarn?
- Welche Vor- / Nachteile ergaben sich aus der Zusammenarbeit?
- Welche inhaltliche Veränderung ergab sich bzgl. der Spezifikation?
- Würdet Ihr lieber allein oder gemeinsam weiterarbeiten?
- Wie würdet Ihr gerne vorgehen bei der Aufgabenstellung eine realistische Spezifikation zu erstellen?
- Welche organisatorischen Aspekte fielen Euch auf?
- Weitere Anmerkungen?

Sichten

- Abstraktion ist wichtig für das Verstehen von komplexen Systemen. Sichten helfen durch ein Ordnungsschema bei der Abstraktion.
- Annahme: Es gibt ein Grundmodell mit Sichten

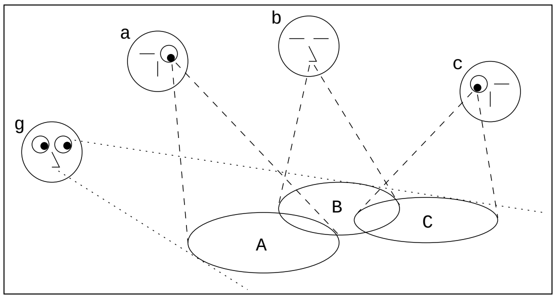


Abbildung 2: Ein Gesamtmodell und drei Sichten

- System zur Vertreibung von Räubern:
  Bremer Stadtmusikanten — Grausame Hexe
  - Katze — glühende Augen und fürchterliche Krallen
  - Hund — scharfe Messer
  - Esel — starker Tritt
  - Hahn — schreckliche Stimme
- Bei Ansätzen häufig: statische, funktionale und dynamische Sicht

Sichten

- Beispiel: Computer Integrated Manufacturing (CIM)
- Bisheriges Problem: Inkonsistenzen
- Lösung: Sichten

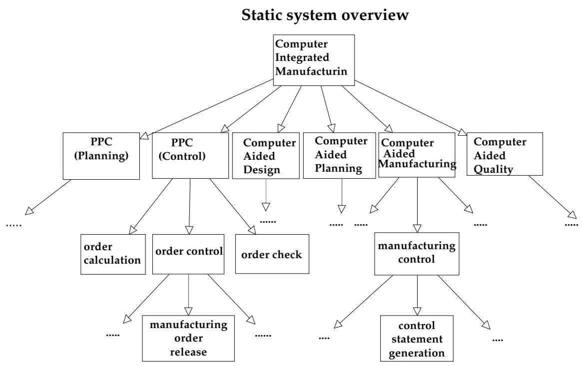


Abbildung 3: Statisches CIM-Modell nach Cronjäger

Sichten

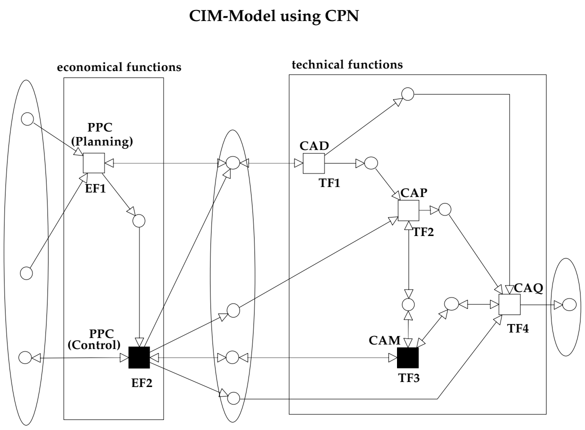


Abbildung 4: CPN CIM-Modell, oberste Ebene

Sichten

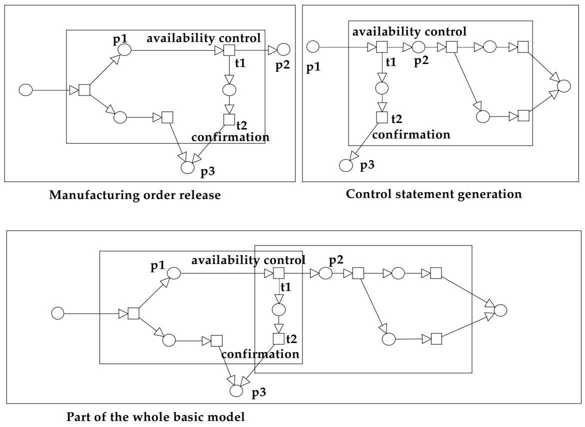


Abbildung 5: CPN CIM-Modell, unterste Ebenen und Gesamtmodell

Sichten

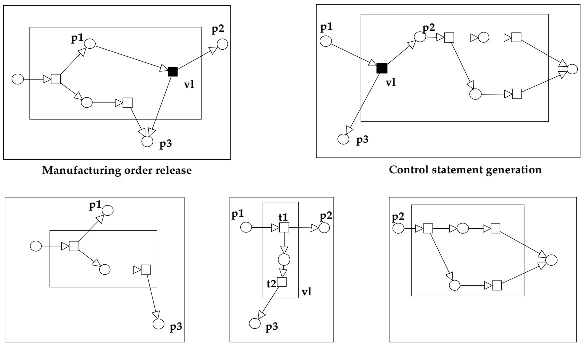


Abbildung 6: CPN CIM-Beispiel, Zerlegung für Sichten

Sichten

- Ergebnisse
  - Abstraktion durch Sichten
  - Statische, dynamische und funktionale Sichten wichtig bei der Spezifikation
  - Modifizierte Strukturierte Analyse anwendbar: Erweitert essentielle Spezifikation
  - Erhöhte Korrektheit der Spezifikationen
  - Besserer Übergang von der Analyse zum Entwurf

Einige Spezifikationsansätze

- Jackson System Development (JSD) Jackson 1983
- Entity-Life Modeling Sanden 1989
- Object-Oriented Design (OOD) Booch 1986
- Object-Oriented Systems Analysis ((OOSA))
  Shlaer und Mellor 1988
- Object-Oriented Requirements Specification Method (OOS)
  Bailin 1989
- Object-Oriented Analysis (OOA)
  Coad und Yourdon 1990, 1991
- Object-Oriented Design (OOD) Booch 1992
- Object-Modeling Technique (OMT)
  Rumbaugh, Blaha et.al. 1991
- Object-Oriented Systems Analysis (OOSA)
  Shlaer und Mellor 1991
- Object-Oriented Analysis and Design (OOA&D)
  Martin und Odell 1992
- Object-Oriented Software Engineering (OOSE) Jacobson 1992
- Object-Oriented Design (OOD) Booch 1994
- Unified Method (UNI) Booch/Rumbaugh 1995
- Unified Method (UML) Booch/Rumbaugh/Jacobson 1996/7
- Object Modelling Language (OML)
  Firesmith/Henderson-Sellers/Graham 1996/7
- Unified Modeling Language (UML)
  Booch/Rumbaugh/Jacobson 1998

Strukturierte Analyse

DeMarco 1978
McMenamin/Palmer Essentielle Systemanalyse 1984
Yourdon Strukturierte Systemanalyse 1989

Die Modelle:

- Datenflußmodel (DFM)
  funktionale Sicht
- Kontrollflußmodel (CFM)
  dynamische Sicht
- Datenmodel (DM)
  statische (Daten-) Sicht

Die Techniken:

- Datenkontextdiagramm (DCD)
- Datenflußdiagramm (DFD)
- Prozeßspezifikation (PSPEC)
- Kontrollkontextdiagramm (CCD)
- Kontrollflußdiagramm (CFD)
- Kontrollspezifikation (CSPEC)
- Materialflußdiagramme (MFD)
- Entity Relationship Diagramme (ERD)
- Endliche Automaten (FA) / (EA)
- Data Dictionary / Repository (DD)

Strukturierte Analyse (SA)

- SA ist eine Methode und ein Ansatz
- Die wesentlichen Techniken sind DFD, CFD und EA.
  Diese stellen die Modelle der Spezifikation dar.
- Das Vorgehen besteht aus vielen Schritten.
- Es wird hier nur ein kurzer Überblick gegeben.
- Die Modellierungsmöglichkeiten beschränken die Möglichkeiten (und die Gedanken) der Modellierenden.

Wichtige Aspekte der DFD

- Terminatoren, Transformatoren, Datenflüsse, Speicher und Beschriftungen bilden die grundlegenden Komponenten.
- Es gibt eine Hierarchie der einzelnen Diagramme.
- Die Datenflüsse zwischen dem System und der Systemumgebung werden dargestellt.
- Das Modell erlaubt es die Datenflüsse innerhalb des Systems zu betrachten.
- Essentielle Funktionen kommunizieren (nur) über Speicher miteinander. Dies kann bei vielen Varianten der SA schon auf der obersten Ebene erkannt werden.

Strukturierte Analyse

Datenkontext-Diagramme Beispiele

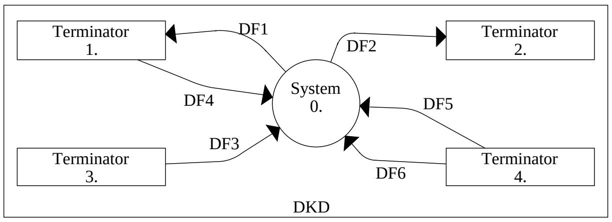


Abbildung 7: Ein Datenkontext-Diagramm

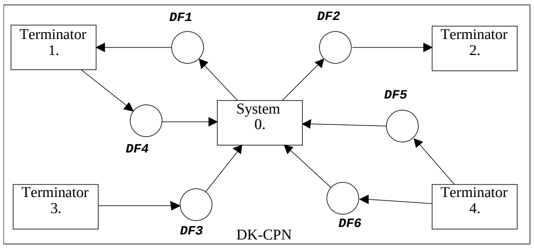


Abbildung 8: Ein Datenkontext-Netz

Datenfluß-Diagamme

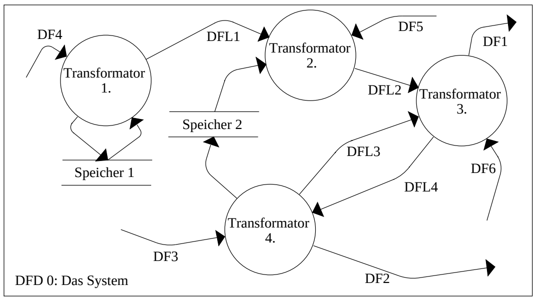


Abbildung 9: Ein Datenfluß-Diagramm

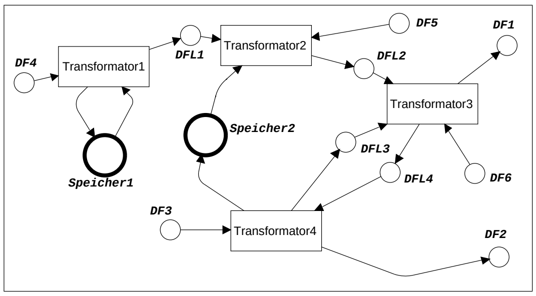


Abbildung 10: Ein Datenfluß-Netz

Prozeß-Spezifikationen

```
PSP 4.: transformer 4.

IF   DF3 > DFL4
THEN DFL3 := DF3;
     DF2  := TRUE;
ELSE DFL3 := DFL4;
     DF2  := FALSE;
END IF;
ADD 1 TO Speicher2;
```

Abbildung 11: Eine Prozeß-Spezifikation

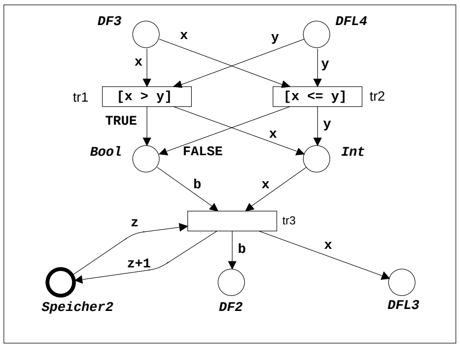


Abbildung 12: Ein Prozeß-Spezifikations-Netz

Strukturierte Analyse

Das Vorgehen

- Finde alle externen Partner.
- Beschreibe die Datenflüsse zwischen dem System und den einzelnen Terminatoren (Systemumgebung).
- Klassifiziere Ereignisse, auf die das System reagieren soll.
- Beschreibe zu jedem Ereignis die zugehörige Funktion.
- Verfeinere die Funktion in Fragmente, so daß die Prozeßspezifikation eine Größe von einer Seite nicht übersteigt.
- Untersuche die Fragmente auf nicht essentielle Komponenten und entferne diese.
- Versuche die Fragmente zu verbinden.
- Entferne alle überflüssigen physikalischen Aspekte aus dem Modell.

Übung

Versuche nun dieses (naive) Vorgehen auf das Aufzugsystem zu übertragen.

Welche weiteren Fragestellungen ergeben sich?

Essentielle Funktionen

Das zu spezifierende System besteht aus essentiellen Funktionen.
Das System kommuniziert mit externen Partnern.
Diese werden durch eingehende Daten oder Signale von außen oder durch einen internen Timer angestoßen. Um alle Funktionen zu erfassen, sind also alle eingehenden Daten und alle systemrelevanten Zeitpunkte zu untersuchen. Im folgenden werden diese als Ereignisse betrachtet.
Eine erste Ermittlung der potentiellen essentiellen Funktionen erfolgt, indem man für die einzelnen eingehenden Daten jeweils die Systemreaktionen beschreibt.
Die so erhaltenen Funktionen lassen sich zusammenfassen. Die Zusammenfassung erfolgt durch die Entfernung der Überschneidungen der Systemreaktionen und der indirekten Mehrfachbehandlung aufgrund der isolierten Behandlung der Daten und Signale.
Hier wird deutlich, daß es wichtig ist die Auslöser von essentiellen Funktionen zu erkennen. Diese Auslöser sind eine Teilmenge der Ereignisse.
Bei einfachen Systemen lassen sich die Auslöser sehr schnell aus der Menge der Ereignisse herauslösen. Ein Hauptproblem ist es zu erkennen, welche Ereignisse zu einem Auslöser gehören.
Hier ist es wichtig zu erkennen, daß bei sehr großen Systemen die nicht-essentiellen Komponenten eines Systems zu immer größerer Unübersichtlichkeit führen. Eine Methode muß auf diesen Umstand eingehen. Die Strukturierte Analyse nach [McMenamin 84] bietet hierfür ein dediziertes Vorgehen an.

Essentielle Funktionen

Wir werden uns hier jedoch damit aus mehreren Gründen nicht beschäftigen:

- Wir können hier keine entsprechend großen Systeme behandeln. Ihr Umfang bewegt sich in vielen Personenjahren.
- Wir halten solche Systeme für nicht effizient handhabbar.
- Prototyping ist heutzutage wesentlich einfacher und ausgereifter als 1984. Dadurch kommt man wesentlich schneller zu brauchbaren Prototypen und damit zu verbindlicheren Aussagen bzgl. eines geplanten Systems.
- Das sogenannte Reengineering verlangt einen totalen Bruch mit alten Systemen. Dabei sind die Inhalte zwar nicht zu vergessen, aber in einer völlig anderen Art und Weise zu hinterfragen. Die Berücksichtigung alter essentieller Funktionen verhindert, daß man zu einer völlig neuen und besseren Organisation kommt.
- etc.

SA bietet für die Entwicklung neuer Systeme (siehe [McMenamin 84]) ebenfalls ein Vorgehen an. Dieses besteht weitgehend aus dem zweiten Teil des Vorgehens bei dem Modellieren eines bestehenden Systems. Es fehlen somit diverse Schritte, in denen die Modelle von nicht-essentiellen Komponenten befreit werden. Bei einer guten Modellierung sollte dies auch nicht notwendig sein.

Essentielle Funktionen

In unserem Beispiel der Aufzugsteuerung haben wir diverse mögliche externe Partner. Von diesen wählen wir ein kleine Menge aus. Für jeden dieser Partner müssen wir ein Kommunikationsprotokoll angeben und die entsprechenden Reaktionen des Systems.
Die externen Partner sind:

- Stockwerkssensor,
- innere Knöpfe,
- äußere Knöpfe,
- Motor,
- Tüeren

Die Auslöser heißen:

- Stockwerksmeldung erfolgt
- Meldung eines inneren Knopfes erfolgt
- Meldung eines äußeren Knopfes erfolgt
- Tür ist geschlossen

Zentrale Ideen der SA

- Kompletter Ansatz
- Trennung WAS – WIE (Essenz)
- Systemabgrenzung
- Verwendung mehrerer Sichten
- Keine echte Kapselung
- Paradigma "Teile und herrsche"

Vorteile von Petrinetzen

Viele der im folgenden aufgeführten Vorteile sind subjektiver Natur und können nicht formal bewiesen werden.
Einige Vorteile von gefärbten Petrinetzen:

- Gefärbte Petrinetze haben eine graphische Repräsentation.
- Die Begriffe Zustand, Aktion und Fluß sind die Basis für viele Arten von Systemen und werden in den gefärbten Petrinetzen als Stellen, Transitionen und Kanten dargestellt.
- Gefärbte Petrinetze haben eine wohldefinierte Semantik, die eindeutig das Verhalten eines jeden gefärbten Petrinetzes definiert.
- Durch die formale Semantik werden Simulatoren und die Grundlagen der formalen Analyse ermöglicht.
- Gefärbte Petrinetze sind sehr allgemein und grundlegend, wodurch eine Vielzahl verschiedener Systeme beschrieben werden können.
- Gefärbte Petrinetze haben sehr wenige, aber mächtige Primitive.
- Die Konzepte sind allgemein in ähnlicher Form bekannt und damit leicht erkennbar. Zudem erleichtert die geringe Anzahl an Primitiven die Entwicklung einer Analysemethode.
- Gefärbte Petrinetze haben eine explizite Beschreibung sowohl von Zuständen als auch Aktionen.

Vorteile von Petrinetzen

- Gefärbte Petrinetze haben eine Semantik, die auf wirklicher Nebenläufigkeit beruht anstatt auf Überlappung (engl. interleaving).
- Gefärbte Petrinetze bieten eine hierarchische Beschreibung(smöglichkeit).
- Gefärbte Petrinetze integrieren die Beschreibungen von Steuerung (engl. control) und Synchronisation mit denen der Datenmanipulation.
- Gefärbte Petrinetze sind stabil gegenüber kleinen Änderungen des modellierten Systems.
- Gefärbte Petrinetze bieten eine interaktive Simulation, wobei die Ergebnisse direkt in den Netzdiagrammen dargestellt werden.
- Gefärbte Petrinetze haben eine große Anzahl formaler Analysemethoden, mit denen Eigenschaften der Netze bewiesen werden können.
- Für gefärbte Petrinetze sind Computerwerkzeuge verfügbar, die die Zeichnung, Simulation und formale Analyse unterstützen.

Ersetzung von Techniken durch Petrinetze

- **Nachteile** der Verwendung von Netzen

  Schlechte **Modellierungsstile**, die durch die Techniken bedingt sind, werden übernommen.
  Die Anzahl der graphischen **Elemente** nimmt (leicht) zu.
  Die Technik ist schwieriger für die **Beteiligten**.
  Eine Unterstützung der Methode durch **Werkzeuge** ist zwingend notwendig.
  Die **Möglichkeiten** der Netze werden nicht voll ausgeschöpft.
  Die **Anwendungskomplexität** verringert sich nicht.

- **Vorteile** der Verwendung von Netzen

  Die **Techniken** lassen sich durch Netze ersetzen.
  Die **Methodik** kann beibehalten werden.
  Die Methodik läßt sich durch Netze **erweitern** und verbessern.
  Es gibt **weniger** unterschiedliche Techniken.
  Die **Inkonsistenzen** zwischen den Modelle verringern sich.
  Die Spezifikationen werden **simulierbar**.
  Eine **automatische Übersetzung** von Techniken in Petrinetze wird durch entsprechende Definitionen möglich.
  Die **Erfahrungen** aus dem Einsatz der jeweiligen Ansätze können übernommen werden.
  Es sind weniger **redundante** Informationen darzustellen.
  Der **Übergang** zum Entwurf und zur Implementation wird erleichtert.

Objektorientierte Systemanalyse (OOA)

- Objektorientierte Ansätze zeigen sich in den letzten Jahren als besonders **erfolgreich**.
- Die Hauptvorteile liegen in der **Verständlichkeit** und Wiederverwendbarkeit der Modelle.
- Die Vor- und Nachteile bzgl. der Verwendung von Petrinetzen entsprechen weitgehend den oben angegebenen.

Objektorientierung und Petrinetze

- Objektorientierte Konzepte sind bei Netzen hauptsächlich für die Beschreibung der **Marken** entwickelt worden.
- Wir haben gezeigt, wie man alle objektorientierten Konzepte direkt über Stellen, Transitionen, Kanten und Marken realisieren kann, ohne auf spezielle **Beschreibungssprachen** zurückgreifen zu müssen.
- Der **Hauptnachteil** dieses Ansatzes ist, daß eine Unterstützung durch ein Werkzeug unbedingt notwendig ist.
- Eine **Einbettung** in eine objektorientierte Umgebung erscheint sinnvoll.
- Ein großer Vorteil ist, dass mit diesem Ansatz die objektorientierten Konzepte um die Nebenläufigkeit **erweitert** werden.

Objekt-orientierte Modellierung

Ziel dieses Abschnittes ist es ein Beispiel für das Vorgehen bei einem objektorientierten Ansatz zu erläutern. Dabei wird auf den Object Modelling Technique (OMT) zurückgegriffen. Es geht dabei um das prinzipielle Vorgehen. Weitere Varianten werden an späteren Terminen vorgestellt.

Literatur:
Rumbaugh, James and Blaha, Michael and Premeralani, William and Eddy, Frederick and Lorensen, William, *Object-Oriented Modeling and Design*, Prentice Hall, Englewood Cliffs, New Jersey 07632, Object-Oriented Modeling and Design, 1991

Der Ansatz OMT besteht aus den folgenden Abschnitten:
1. Analyse
2. Systementwurf
3. Objektentwurf und
4. Implementation

OMT erstellt im wesentlichen drei Modelle:
1. Das Objekt Modell
2. das funktionale Modell und
3. das dynamische Modell

Bau eines Objektmodells

- Objekte und Klassen identifizieren
- Data Dictionary vorbereiten
- Assoziationen incl. Aggregationen zwischen Objekten identifizieren
- Attribute von Objekten und Verknüpfungen identifizieren
- Objektklassen mit Hilfe von Vererbung organisieren und vereinfachen.
- Sicherstellen, daß für wahrscheinliche Anfragen Zugriffspfade existieren.
- Das Modell überarbeiten und verfeinern.
- Klassen zu Modulen gruppieren.

Analyse

[Rumbaugh 93] S. 181: Beim ersten Schritt des OMT-Ansatzes, der Analyse, ist es das Ziel ein präzises, kompaktes, verständliches und korrektes Modell der realen Welt zu entwickeln. Aufgabe der OOA ist es, verständliche Modelle zu erstellen, dabei sind wichtige Eigenschaften abstrakt darzustellen und kleinere Details außer acht zu lassen. Das Analyseergebnis besteht darin, das Problem in Vorbereitung auf den Entwurf zu verstehen. Die Zwecke des Analysemodells: Abklärung der Anforderungen, Verständigungsgrundlage für Auftragegeber und Softwareentwickler sowie Bezugsrahmen für Entwurf und Implementation.

Aus der Problembeschreibung wird das Analysemodell, bestehend aus Objektmodell, Dynamischem Modell und Funktionalem Modell, durch den Prozeß der Analyse erstellt. Nicht alle drei Teilmodelle sind gleich wichtig für jedes Problem. Die Modelle werden iterativ erstellt.

Die Problembeschreibung ist intensiv zu hinterfragen und auf die tatsächlichen Anforderungen hin zu untersuchen. Entwurfsentscheidungen sind, soweit es möglich ist, zu vermeiden.

Beginn mit Objektmodell

- Identifikation der Klassen und Assoziationen
- Hinzufügen der Attribute
- Verbinden der Klassen durch Vererbung
- Operationen erst als Nebenprodukt der anderen beiden Modelle.

Bau eines Objektmodells

1. Objekte und Klassen identifizieren
   - Objekte sind physikalische Entitäten
   - Nur Klassen mit Bedeutung
   - Substantive liefern geeignete Klassen
   - Nur geeignete Klassen behalten
2. Data Dictionary vorbereiten
   - Beschreibung jeder Objektklasse
   - Beschreibung aller "Dinge"
3. Assoziationen incl. Aggregationen zwischen Objekten identifizieren
   - Abhaängigkeiten zwischen zwei und mehr Klassen beschreiben
   - Assoziationen entsprechen Verben
   - Nur geeignete Assoziationen beibehalten
     - Eliminierte Klassen
     - Irrelevante oder Implementationsassoziation
     - Aktionen
     - Ternäre Assoziationen
     - Abgeleitete Assoziationen
     - Falsch benannte Assoziationen
     - Rollennamen
     - Qualifizierte Assoziationen
     - Mulitplizität
     - Fehlende Assoziationen

4. Attribute von Objekten und Verknüpfungen identifizieren
   - Attribute entsprechen meist Substantiven, an die ein Genitiv anschließt.
   - Attribute fehlen häufig in den Problembeschreibungen
   - Nur geeignete Attribute beibehalten
     - Objekte
     - Qualifikationsangaben
     - Namen
     - Identifikatoren
     - Verknüpfungsattribute
     - Interne Werte
     - Details
     - Einander widersprechende Attribute
5. Objektklassen mit Hilfe von Vererbung organisieren und vereinfachen.
   - Vererbung einsetzen, um gemeinsame Strukturen zu nutzen
   - Zwei Richtungen sind möglich:
     (a) Generalisierung gemeinsamer Aspekte bestehender Klassen in eine Oberklasse (bottom up)
     (b) Verfeinerung bestehender Klassen in spezialisiertere Unterklassen (top down)
   - Vererbung finden, durch suchen nach ähnlichen Attributen, Assoziationen oder Operationen
   - Spezialisierungen ergeben sich oft aus dem Anwendungsbereich
   - Mehrfachvererbung ist sparsam einzusetzen

   - Bei gleichen Assoziationsnamen versuche die Klassen zu generalisieren
6. Sicherstellen, daß für wahrscheinliche Anfragen Zugriffspfade existieren, dabei sollte man sich an sinnvollen Fragen orientieren und auf die Komplexität in Modell und Wirklichkeit achten.
7. Das Modell überarbeiten und verfeinern.
   - Fast immer sind mehrere Durchgänge notwendig
   - Hinzufügen oder Spalten von Klassen, je nach Bedarf
   - Unnötige Klassen entfernen
   - Erfahrung nutzen, um mehrere Schritte zur gleichen Zeit zu machen und unterschiedliche Geschwindigkeiten innerhalb des Projektes zu ermöglichen
     aber: Anfänger sollten sich sehr stark an die Vorgaben halten!
8. Klassen zu Blättern und Modulen gruppieren.
   - Ein Modul ist eine Menge von Klassen (ein oder mehrere Blätter)
   - Aufteilung in Blätter oder Module an Klassen, die nur mit wenigen (zwei) anderen Klassen verbunden sind.
   - Anzahl der "Brücken" möglichst klein halten
   - Sternmuster, mit Kernmodul für die Globalstruktur der Klassen und Module für die Verfeinerungen der einzelnen Klassen der obersten Ebene
   - Wiederverwendung von alten Modulen, wenn möglich

Dynamische Modellierung

1. Beginnen mit der Suche nach Ereignissen, danach zusammenfassen zulässiger Ereignisfolgen für jedes Objekt
2. Dynamische Modelle wichtig für interaktive Systeme
3. Echtzeitaspekte werden hier nicht behandelt
4. Szenarien von typischen Interaktionssequenzen vorbereiten
   - Schreiben von mehreren typischen Dialogen (Benutzer und System)
   - Dadurch erhält man die wichtigsten Interaktionen, externe Anzeigeformate und Kommunikationsflüsse
   - Erst die normalen Fälle, dann Sonderfälle
   - Ein Szenario ist eine Folge von Ereignissen
   - Ein Ereignis tritt immer dann auf, wenn Informationen zwischen einem Objekt im System und einen äußeren Agenten ausgetauscht werden. Informationswerte sind Parameter des Ereignisses.
   - Zuweisen der Ereignisse zu Handlungs - und Zielobjekten
5. Scharfe Trennung von Anwendungslogik und Benutzerschnittstelle

Dynamische Modellierung (Forts.)

6. Ereignisse (zwischen Objekten) identifizieren
   - Gruppieren von gleichen Ereignissen
   - Trennen von Ereignissen, die sich auf den Kontrollfluß auswirken.
   - Zuweisen der Ereignisse zu Handlungs - und Zielobjekten
   - Einen Ereignispfad für jedes Szenario vorbereiten
   - Ereignispfad als geordnete Liste tabellarisch angeordneter Ereignisse zwischen verschiedenen Objekten darstellen.
   - Ereignisflußdiagramm für die Ereignisse zwischen einer Gruppe von Klassen (Modulen)
7. Ein Zustandsdiagramm entwicklen
   - Für jedes Objekt mit nicht-trivialem Verhalten ein Zustandsdiagramm erstellen, das die Ereignisse zeigt, die das Objekt empfängt und sendet.
   - Abarbeiten eines Szenarios
   - Einarbeiten der anderen Szenarien
   - Komprimieren der Diagramme, durch Variablen (Daten)
8. Ereignisse zwischen den Objekten vergleichen, um die Konsistenz zu verifizieren.
   - Synchronisationsprobleme
   - Vollständigkeit

Funktionale Modellierung

Das funktionale Modell zeigt, wie Werte berechnet werden, ohne Reihenfolgen, Entscheidungen oder Objektstrukturen zu berücksichtigen. Es zeigt, welche Werte von welchen anderen Werten abhängen, und beschreibt die Funktionen, die sie verbinden [Rumbaugh 94, S.217].

Die Prozesse in einem DFD entsprechen Aktivitäten oder Aktionen in den STDs der Klassen.

Die Flüsse in einem DFD entsprechen Objekten oder Attributwerten in einem Objektdiagramm (OD).

DFDs sollten nach STDs und ODs erstellt werden.

1. Ein- und Ausgabewerte identifizieren, indem die Parameter von Ereignissen betrachtet werden.
2. Datenflußdiagramme erstellen, die funktionale Abhängigkeiten zeigen; Schichtenweise Erstellung der Diagramme; von Ausgabewerten der Funktionen ausgehen und Eingaben bestimmen.
3. Funktionen beschreiben
4. Einschränkungen identifizieren
5. Optimierungkriterien spezifizieren

Operationen hinzufügen

Analyse überarbeiten

Objektorientierte Systemanalyse (OOA)

Use Cases