8 Modellierungshilfsmittel

Abfolge:

1. Spezifikation
2. Entwurf
3. Codierung

Übergang kann fliessend sein.

Aber: In konkretem Projekt zieht man Grenzen

8.1 Spezifikationshilfsmittel

Data Dictionary: Wie sieht die Ein- und Ausgabe genau aus

Datenflussdiagramm: Grobe Zusammenhänge?

Endliche Automaten: Was genau läuft ab?

Logik: Viefältige Anwendungen. Beispiel: Formulierung von Klasseninvarianten. Vor- und Nachbedingungen, etc.

Bereichsspezifisch: Viele Möglichkeiten

Relationale Datenbanken: Entity Relationship Diagramme, Wie hängen Relationen zusammen

Verteilte Systeme: Petrinetze, ...

Echtzeitsysteme: Zeitanforderungen modellieren

Expertensysteme: Regeln

8.1.1 Data Dictionary

Definition legaler Eingabe- und Ausgabedateien durch Grammatiken.

- Reguläre Ausdrücke sind kompakt
- Reguläre Erweiterungen [] =Option, $()^+$ mindestens einmal, ...
- Kontextfreie Grammatiken erlauben Klammerstrukturen
- Syntaxdiagramme,
- ...

Probleme:

- Anwendung für Spezifikation von Mausereignissen? (Nebenläufigkeit)
- Modellierungssprache und Einlesewerkzeuge abstimmen! (von Hand, Scannergenerator, Parsergenerator, Bibliothek, ...)

8.2 Entwurfswerkzeuge

- Abstrakte Datentypen beschreiben
  - Algebraische Spezifikation
  - Programmiersprachennah
  - Grafische Darstellung
- Klassenhierarchien
- Strukturdiagramme

8.2.1 Klassenhierarchien

- Veinfachte grafische Darstellung von Klassenname, members, member functions
- Verwandtschaftsbeziehungen
- abstrakt/konkret
- Referenzpfeile (mit Kardinalitäten)

8.2.2 Strukturdiagramme

Grundidee: Schnittstelle abstrahiert vom „Wie“ der Implementierung.

Funktion/Prozudur: Kleinste Abstraktionseinheit

Modul: Zusammenfassung zusammengehöriger Funktionen

Exportschittstelle: Zur Verfügung gestellte Funktionalität

Importschnittstelle: Von anderen Modulen benötigte Unterstützung.

Hierarchischer Aufbau.

8.3 Struktur von Modulbeziehungen

Baum: Klassischer Top-Down-Entwurf

DAG: (directed acyclic graph), Graph ohne gerichtete Kreise. Modulwiederverwendung. Problem: Top-Down-Entwurf macht es schwer Wiederverwendungsmöglichkeiten zu finden.

mit Kreisen: verpönt. Aber immer vermeidbar?

Schichten: Anordnung der Module nach natürlicher Abstraktionsebene.

- Innerhalb einer Schicht i. allg. freie Struktur, einschließlich Kreisen.
- Schicht i darf ansonsten Schichten $j < i$ aufrufen.
- Lineare Ordnung: Schicht i darf nur Schicht $i - 1$ aufrufen. (Beispiel ISO/OSI)

8.3.1 Methodischer Ansatz beim Strukturierten Entwurf

1. Datenflussdiagramm
2. Zentrale Funktionen ermitteln
3. Erstes Strukturdiagramm
4. Verfeinern
5. Überprüfen

8.4 Hilfsmittel zur Algorithmenbeschreibung

- Flußdiagramme. (Ggf. bei kompliziertem Ablauf sonst wenig gebräuchlich. (Goto läßt grüßen.))
- Struktogramme. Übersichtlich aber aufwendig.
- Pseudocode. Ggf. ergänzt durch mathematische Notation

(* Drucke Primzahlen bis $n \geq 2$ (langsam) *)
print 2
foreach $\{3 \leq p = 2j+1 \leq n\}$ **do**
    if $\forall\{3 \leq t = 2j+1 \,|\, t^2 \leq p\} : t \not\equiv 0 \bmod p$ **then**
        print p