UML-Seminar

ROOM: Eine Methode zur Entwicklung von Echtzeitsystemen

Einleitung

- Komplexität der Systeme wächst kontinuierlich
- Anforderungen an Entwicklungszeiten werden immer größer
- Entwicklung von Echtzeitsystemen bringt eine große Anzahl von Fehlerquellen mit sich
- Entwicklung von ROOM (Real-Time Object-Oriented Modelling) Anfang der 90er bei Bell-Northern Research
- ROOM ist eine kompakte Menge sorgfältig ausgewählter Konzepte, zugeschnitten auf eine spezifische Klasse von Systemen

Komponenten

- eine Modellierungssprache (im Ggs. zur Implementierungssprache)
- eine Modellierungsheuristik
  - informelle Richtlinien für die Benutzung der Modellierungssprache
- ein Rahmenwerk zur Organisation und Durchführung der Entwicklungsarbeit
- erhöhte Chancen auf einen positiven Abschluss des Projektes

Echtzeitsysteme

- es gibt keine allgemeingültige Definition (Mikrocontroller bis Kommunikationsnetzwerke)
- Echtzeitsysteme bestehen aus einer Kombination von Hard- und Software, Sensoren und Effektoren
- passen ihr Verhalten an Veränderungen der äußeren Umgebung an

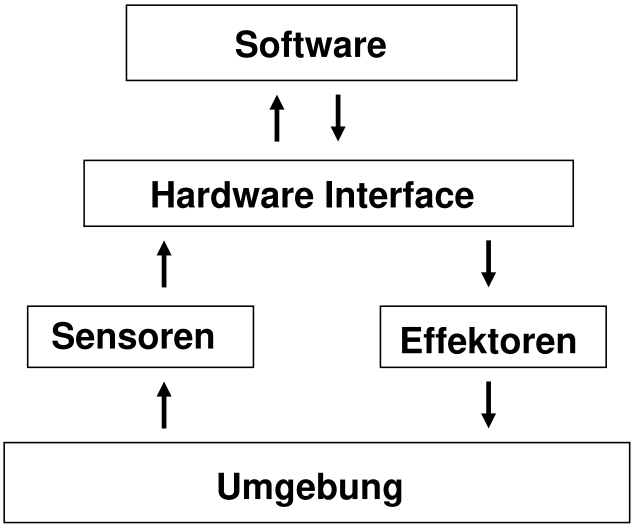

Abbildung 1: Elemente eines Echtzeitsystemes

Charakteristiken von Echtzeitsystemen 1

Zeitgarantien (Timeliness):

- service time: benötigte Zeit, um eine Ausgabe zu den Eingaben zu berechnen (oft deterministisch und vorhersagbar)
- latency: Intervall zwischen dem Zeitpunkt der Eingabe und dem Beginn der Berechnung der Ausgabe (oft schwankend, da sowohl fixe als auch variable Elemente)
- die Summe aus service time und latency ergibt die Reaktionszeit (reaction time)
- bei sog. harten Echtzeitsystemen ist das Nichterfüllen einer deadline unakzeptierbar (z.B. lebenskritische Systeme)
- Problem: den Zeitverbrauch von Designentscheidungen hinreichend genau zu ermitteln

Anforderungen an die Modellierungssprache:

- Konzepte der absoluten Zeit und Zeitintervalle

- abschätzbare Reaktionszeit
- Erstellen von Prototypen des Systemes, um das Zeitverhalten in der Implementierungsumgebung zu messen

Charakteristiken von Echtzeitsystemen 2

dynamische interne Struktur:

- dynamische Veränderungen in der Umgebung erfordern eine Rekonfiguration des Systemes
- dynamisches Kreieren und Zerstören von Komponenten und Beziehungen zwischen den Komponenten des Systemes

Anforderungen an die Modellierungssprache:

- die internen Strukturen der erstellten Modelle müssen zur Laufzeit modifizierbar sein

Charakteristiken von Echtzeitsystemen 3

Reaktivität (Reactiveness)

- das System hat keine Kontrolle über die Reihenfolge, in der die Eingaben aus der Umgebung erfolgen (Nichtdeterminismus)
- die Antwort des Systemes hängt von den vorangegangenen Eingaben und dem Zeitpunkt der Eingabe ab (Zustandsabhangigkeit)

Charakteristiken von Echtzeitsystemen 4

Nebenläufigkeit (Concurrency)

- der Zustand des Systemes wird durch mehrere Prozesse transformiert
- dynamische Beziehungen zw. den Prozessen machen Wechselwirkungen unübersichtlich
- – Synchronisierung: Zeitpunkt der Ausführung wird durch Zustand anderer Prozesse bestimmt
  – Kommunikation: Informationsübertragung zwischen Prozessen
- mehrere Prozesse, Ein- und Ausgaben werden in einem sog. scenario zusammengefasst (Visualisierung durch message sequence charts), um funktionale Anforderungen auszudrücken
- Konflikte können durch die nebenläufigen und asynchronen Eigenschaften der Umgebung auftreten

Anforderungen an die Modellierungssprache:

- Modellierung der Wechselwirkungen zwischen den Prozessen
- Modellierung von abstrahierenden scenarios
- Unterstützung bei dem Lösen der durch die Nebenläufigkeit entstehenden Probleme

Charakteristiken von Echtzeitsystemen 5

Verteilung (Distribution)

- stark erhöhte Komplexität des Systemes
- Verlust oder Korruption von Nachrichten (kein zuverlässliges Protokoll bei nicht garantierter Kommunikation)
- verzögerte und variable Kommunikationszeiten
- Teilausfälle des Gesamtsystemes
- eine gemeinsame Sicht des Systemzustandes kann nicht gleichzeitig von allen Teilen erreicht werden

Anforderungen an die Modellierungssprache:

- Modellierung des Systemes als Zusammenschluß verschiedener, spezialisierter Teile
- Explizite Modellierung der Wechselwirkungen zwischen diesen Teilen

Komplexität

- Hauptproblem ist die Komplexität des zu modellierenden Systemes
- Abstraktion in Form von:
  - Rekursion
  - inkrementelles Modellieren
  - Wiederverwendung

  Anforderungen an die Modellierungssprache:
  - Unvollständige Modelle müssen Eigenschaften des Systemes erkennen lassen
  - Gemeinsamkeiten zw. Teilen müssen ausnutzbar sein und konsistent gehalten werden
  - Abstraktionsmechanismen schaffen Ausgewogenheit zw. Effizienz und Nutzen (Verlässlichkeit, Kontrollierbarkeit und Wartbarkeit) des Systemes

ROOM Schlüsselelemente 1

Unregelmässigkeitstypen bei der Modellierung durch fehlende formale Verbindungen:

- Bereichs-U. (scope): zw. den Repräsentationen verschiedener Detaillevel erzeugt
- semantische U.: zw. den Repräsentationen verschiedener Arten gleichen Detaillevels
- Entwicklungsphasen U.: zw. den Anforderungs-, Design- und Implementierungsrepräsentationen
- rush-to-code Syndrom und verschiedene Arten von Modellierungsabstraktionen führen zu semantischen U.
- low-level Details führen zu Bereichs-U., da oft keine Verbindung zu der Repräsentation im Modell vorhanden ist

Die Bildung dieser U. soll in ROOM durch

- das operationelle Herangehen (operational approach)

- von Entwicklungsphasen unabhängige Modellierungsabstraktionen
- das Objekt-Paradigma

verhindert werden.

Das operationelle Herangehen

- Verbindung zw. low- und high-level details wird formalisiert
- Anforderungs- und Designmodelle werden in einer ausführbaren Modellierungssprache ausgedrückt
- state machines bieten diese Ausführbarkeit und Beobachtbarkeit der Ergebnisse sowie eine kompakte Darstellung höherer Abstraktionslevel
- Kompilierbarkeit und Verifizierbarkeit der Modelle führt zu schnellen Ergebnissen und Einsichten

Ausführbare Modelle müssen

- auf der Entwicklungs- und Implementierungshardware laufen
- Sensoren und Effektoren simulieren können
- mit einer simulierten Echtzeituhr laufen
- architekturelle sowie detaillierte Eigenschaften des Systems verifizierbar machen