Struktur der UML-Spezifikationen

Udo Kelter

24.06.2010

Zusammenfassung dieses Lehrmoduls

Die UML-Spezifikationen bestehen i.w. aus 4 umfangreichen Hauptdokumenten (UML Infrastructure, UML Superstructure, Diagram Interchange, Object Constraint Language), deren Struktur sehr komlex ist. Dieses Lehrmodul liefert eine Einführung die Struktur. Zentral ist das Superstructure-Dokument, in dem die Modellelemente der UML definiert werden. Hierbei werden intensiv Klassendiagramme als Metamodelle eingesetzt; die Struktur der Metamodelle wird im Infrastructure-Dokument definiert, wiederum mit Hilfe von Klassendiagrammen. Zusätzlich verkompliziert wird die Struktur der Spezifikationen durch Compliance-Levels, die partielle Implementierungen ermöglichen.

Vorausgesetzte Lehrmodule:

obligatorisch:
- Die Unified Modelling Language (UML) Version 2
- Modellgetriebene Software-Entwicklung
- Metadaten

Stoffumfang in Vorlesungsdoppelstunden: 0.4

Inhaltsverzeichnis

1 Ziele der UML-Spezifikation

Struktur der UML-Spezifikation ist kompliziert, weil viele überlappende Dinge zugleich spezifiziert werden:

"repository semantics": UML-Spezifikation spezifiziert in erster Linie wesentliche Aspekte von *Modellierungswerkzeugen* bzw. allg. Systemen, die Modelle verwalten oder handhaben; hier zu spezifizieren:

- konzeptueller Inhalt von Modellen diverser Typen
- teilweise die graphische Darstellung von Modellelementen

"run-time semantics": UML mutiert immer mehr zu einer graphischen Programmiersprache

- → UML-Spezifikation spezifiziert eine vage operationale Semantik mancher Modellelemente, also wie das modellierte System sich verhalten soll;
- Vorsicht: Systemwechsel und Wechsel der Bedeutungsebenen!
- run-time semantics kann man auch als Beschreibung ansehen, wie Modelle in Code übersetzt oder interpretatiert werden sollen ...

Beispiele von Übersetzungen:

- Übersetzung von Aktivitätsdiagrammen in Quellcode
- Generierung von Klassenrümpfen

aber: ".. not every concept in UML models a run-time phenomenon..." (s. UML Superstructure, 6.3)

Profile und domain-specific languages (DSL): UML möchte anpaßbar / erweiterbar sein:

- vorhandenen Modelle einschränken und ggf. versehen mit speziellen Bedeutungen

- Erweiterung um neue Modellelemente und neue Diagrammtypen

Benachbarte Probleme:

- Transformation von Modellen in Modelle (z.B. beim MDA-Ansatz)
- Austausch von Modellen zwischen verschiedenen Werkzeugen, Definition von Transportdateiformaten

bedingen alle, den konzeptuellen Inhalt von Modellen und teilweise das Layout zwischen Werkzeugen / Werkzeugfunktionen austauschen zu können

2 Modellbegriff der UML

Diagramme vs. Modelle:

- **Modell** = Gesamtdarstellung des modellierten Systems;
- **Diagramm** = Auswahl ("Zeichenfläche"), die bestimmte Modellelemente und deren Verbindungen (graphisch) zeigt
- gleiches Modellelement kann in mehreren Diagrammen vorkommen (als Spiegelung),
  sogar in unterschiedlichen Darstellungen ("Sichten")
- Diagramme sind für Modelltransformationen irrelevant

Diagrammtypen in der Praxis / beim Editieren:

- Diagrammtypen: Festlegung pro Diagrammtyp, welche Arten von Modellelementen darin erlaubt sind
- ausschließliche Verwendung typisierter Diagramme
- Editoren kontrollieren Verwendung der Modellelemente

Diagrammtypen in der UML:

- definiert 13 Diagrammtypen vor, bleibt aber offen für benutzerdefinierte Diagrammtypen
- daher generell keine Restriktionen für die erlaubten Modellelemente in bestimmten Diagrammtypen
- Diagramme sind keine Modellelemente!

3 Struktur der UML-Spezifikationen

Hauptdokumente:

UML Superstructure [UMLSuS] beschreibt für alle Modellelementtypen:

- deren konzeptuelle Repräsentation selbstreferentiell mit Klassendiagrammen (Metamodelle!)
- ggf. deren graphische Darstellung
- ggf. deren Bedeutung, vor allem durch Prosa

UML Infrastructure [UMLInS] beschreibt:

- Grundbegriffe wie Pakete, Klassen usw.
- insb. alle Modellelemente, die im Superstructure-Dokument für die Metamodelle gebraucht werden (also Meta-Metamodelle!)

Object constraint language:
Sprache, in der zusätzliche Bedingungen formuliert werden können, die Instanzen von Klassendiagrammen erfüllen müssen

Diagram interchange:
behandelt Probleme beim Austausch von Diagrammen (!) zwischen UML-Werkzeugen, also auch Layoutdaten, Behandlung von Koordinaten, ...

4 UML Infrastructure

definiert insb. die Metamodelle, die in der UML Superstructure benutzt werden, um die Struktur von Modellelementen zu spezifizieren

Auszug aus der Gliederung von [UMLInS]:

...
7 Language Architecture
8 Language Formalism

9 Core::Abstractions (u.a. Parameter, Classifiers, Elements, ...)
10 Core::Basic
10.1 Types Diagram
10.2 Classes Diagram
10.3 DataTypes Diagram
10.4 Packages diagram

11 Core::Constructs
11.1 Root diagram
11.2 Expressions Diagram
11.3 Classes Diagram
11.4 Classifiers Diagram
....
11.8 Operations Diagram
11.9 Packages diagram
12 Core::PrimitiveTypes
13 Core::Profiles

Aufgabe: vgl. die Modellierung von Klassen in
10.2, S.93[105] für Meta-Metamodelle
11.3, S.109[121] für Metamodelle

5 UML Superstructure

5.1 Inhaltlicher Aufbau

Inhaltlicher Aufbau von [UMLSuS] (V.2.3) *orientiert an Modellelementen*, gruppiert nach Themenbereichen, (nicht Diagrammtypen); Auszug aus der Gliederung:

...
Part I - Structure

7. Classes	(55 Modellelementtypen)
8. Components	(5 MET.)
9. Composite Structures	(15 MET.)
10. Deployments	(12 MET.)

Part II - Behavior
11. Actions (56 MET.)
12. Activities (52 MET.)
13. Common Behaviors (31 MET.)
14. Interactions (31 MET.)
15. State Machines (16 MET.)
16. Use Cases (6 MET.)

Part III - Supplement
17. Auxiliary Constructs
18. Profiles

Part IV - Annexes Annex A: Diagrams (-typen!)

Konzeptueller Inhalt von Modellelementen wird vielfach durch Klassendiagramme (= Metamodelle) dargestellt → Bezug auf die UML Infrastructure

Hierbei benutzte Metamodelle werden großenteils aus der Infrastructure als Pakete *importiert*!
(obwohl sie dort Meta-Metamodelle sind und hier nur Metamodelle)

5.2 Teilmengenbildung

Ziel: mehr oder weniger vollständige Implementierungen ermöglichen, die sich trotzdem standardkonform nennen können
s. Kap. 2 Conformance (S. 1 [17])

Language Units: Grobzerlegung in zusammenhängende Bereiche; werden weiter zerlegt in Pakete

Compliance Levels: Level 0 (L0) ... Level 3 (L3): zunehmende Abdeckung der UML

"The contents of language units are defined by corresponding top-tier packages of the UML metamodel, while the contents of their various

increments are defined by second-tier packages within language unit packages."

Übungsaufgaben:

1. Versuchen Sie, das Klassendiagramm in Bild Figure 16.2 - "The concepts used for modeling use cases" auf S. 604 [619] der UML Superstructure Version 2.3, das den konzeptuellen Inhalt von Anwendungsfällen spezifiziert, zu verstehen. (Viel Glück!)
2. Planen Sie ein Projekt, in dem Sie unter Einsatz des EMF einen UML2.0-konformen grapischen Editor für Anwendungfalldiagramme realisieren.

auch ansehen:

- in [UMLInS]
  9.17.1 DirectedRelationship
  9.17.2 Relationship
- Bedeutung der Constraints "subsets"
  [UMLInS] 6.2.1 Diagram format, S.7 bzw.
  [UMLSuS] 6.2.4 Diagram format, S.19
- Bedeutung von Ableitungen
- generell: [UMLSuS] 6.2.4 Diagram format

Literatur

[UML20I] Unified Modeling Language: Infrastructure, Version 2.0; OMG, Doc. formal/05-07-05; 2006

[UML20S] Unified Modeling Language: Superstructure, Version 2.0; OMG, Doc. formal/05-07-04; 2006

[UML20DI] Diagram Interchange, Version 1.0; OMG, Doc. formal/06-04-04; 2006

[UML20OCL] Object Constraint Language, Version 2.0; OMG, Doc. formal/06-05-01; 2006