

Inhalt

Editorial

1 Motivation und Zielsetzung 1

- 1.1 Erstellung von Diagramm-Editoren 1
 - 1.1.1 Komplexität des Systems 3
 - 1.1.2 Benutzer-Entwickler-Kommunikation 4
 - 1.1.3 Dauer der Entwicklungszyklen 6
 - 1.1.4 Bindung zwischen Diagramm- und
Nutzinformationen 6
 - 1.1.5 Datenbankanbindung 6
- 1.2 Modellbildung 7
- 1.3 Ansatz und Inhalt der Arbeit 9
- 1.4 Typographie und Notationen 12

2 Modellbildung und objektorientierte Systeme 13

- 2.1 Modellbildung in der Softwareentwicklung 13
 - 2.1.1 Durchgängigkeit im Softwareentwicklungsprozeß 13
- 2.2 Objektorientierte Modelle 15
 - Objekte 15
 - Klassen 15
 - Attribute 16
 - Operationen, Botschaften und Protokolle 16
 - Beziehungen 16
 - Datenkapselung 18
 - Vererbung 18
 - Polymorphismus und Operationenprotokolle 19
- 2.3 Objektorientierte Programmiersprachen 20
- 2.4 Metamodelle 21
- 2.5 Modellrepositories 23

- 2.6 Entwurfsmuster 26
- 2.7 Objektorientierte Frameworks 27

3 Objektorientierte Datenbanksysteme 29

- 3.1 Zielsetzung 29
- 3.2 Der ODMG-Standard 30
 - 3.2.1 Object Model 30
 - 3.2.2 ODL und OIF 31
 - 3.2.3 OQL 32
 - 3.2.4 Die C++-Sprachbindung 33
 - 3.2.5 Die Java-Sprachbindung 34

4 Graphiksysteme 37

- 4.1 Graphikausgabe 37
 - Koordinatensysteme 38
 - Erzeugungsalgorithmen 38
 - Farbmodelle 38
 - Textdarstellung 38
- 4.2 Layoutkonzepte 39
- 4.3 Ereignisbehandlung 40
 - Ereignisabfrage und -verarbeitung 40
 - Eingabetechniken für graphische Elemente 40
- 4.4 Anbindung von Anwendungen 40
 - Das MVC-Modell 41
- 4.5 Hierarchische Graphiksysteme 43
- 4.6 Objektorientierte Graphiksysteme 44
- 4.7 Graphikbibliotheken 48
 - Das Andrew-Toolkit 48
 - X-Toolkit (Xt) 48
 - InterViews und Fresco 48
 - ET++ 49
 - AWT 50

Weitere Bibliotheken 50
Vergleich der Bibliotheken 50

5 Empirische Analyse verfügbarer Diagramm-Editoren 53

- 5.1 Vergleichskriterien 53
 - Allgemeine Editorfunktionalität 53
 - Graphische Darstellung von Objekten 53
 - Graphische Darstellung der Verbindungen 54
 - Zugriff auf Nutzinformationen 54
 - Konsistenzsicherung 54
 - Reverse Engineering 54
 - Nichtgraphische Funktionalität 54
 - Datenhaltung 54
- 5.2 Cadence 55
- 5.3 Paradigm Plus 57
- 5.4 Microsoft Project 59

6 Verwandte Ansätze 61

- 6.1 Frameworks für Diagramm-Editoren 61
 - 6.1.1 Unidraw 61
 - 6.1.2 vis-a-vis 62
- 6.2 Visuelle Programmiersprachen und Entwicklungssysteme 63
 - 6.2.1 Positionsorientierte Grammatiken und Compiler für VPLs 63
 - 6.2.2 VLCC 66
 - 6.2.3 DiaGen 68
 - 6.2.4 Agentsheets 69
 - 6.2.5 Objection 71
 - 6.2.6 Paradigm Plus 72
- 6.3 Modellgetriebene Generierung von Standard-Dialogoberflächen 73

- 6.3.1 JANUS/JADE 73
- 6.3.2 GENIUS 74
- 6.3.3 UIDE 75
- 6.3.4 HUMANOID 75
- 6.3.5 TADEUS 75

7 Modellgetriebene Generierung von Diagramm-Editoren 77

- 7.1 Graphische Repräsentation und
Bearbeitung objektorientierter Systeme 78
 - Abbildbarkeit auf Objektebene 78
 - Abbildbarkeit auf Modellebene 78
 - Eingangsinformationen für einen
Generierungsprozeß 80
 - Vollständigkeit der Editor-Funktionalität 80
- 7.2 Systemanforderungen und -komponenten 81
- 7.3 Aufbau und Eigenschaften des generierten Editors 84
- 7.4 Abbildung von Klassen 86
 - 7.4.1 Objektsymbole 86
 - 7.4.2 Kopieren und Löschen von Symbolen 88
 - 7.4.3 Textelemente und nichtlineare Dehnung 88
 - 7.4.4 Popup-Menüs für Objektsymbole 89
 - 7.4.5 Der Symboleditor 90
- 7.5 Abbildung von Beziehungen 91
 - Abgrenzung gegen constraint-basierte Ansätze 92
- 7.5.1 Verbindungskonstellationen 93
 - Rechteckpfadverbindungen 93
 - Punktkoinzidenzverbindungen 95
 - Einfügeverbindungen 97
- 7.5.2 Der Editor für Verbindungskonstellationen 98
- 7.5.3 Interaktionsmodelle 100
 - Weitere Eigenschaften von Beziehungen 101
 - Gekoppelte Bearbeitung von Verbindungen und
Objekten 101
 - Assoziationsmenü 101

	Feste (*:1)-Assoziationen	102
	Optionale (*:1)-Assoziationen	103
	Verbinden über Drag and Drop	104
	Optionale (*:n)-Assoziationen	105
	Feste (*:1)-Aggregationen	106
	Optionale (*:1)-Aggregationen	108
	Optionale (*:n)-Aggregationen	108
7.5.4	Typsicherheit	109
7.6	Abbildung von Attributen	112
7.6.1	Bearbeitungsdialoge	112
7.6.2	Direkter Zugriff über Textelemente	114
	Graphische Darstellung von Attributzuständen	117
7.7	Abbildung von Operationen	118
7.8	Abbildung von Vererbungsbeziehungen	120
7.9	Abbildung von Beziehungsklassen	122
	Zugriff auf das Beziehungsobjekt	124
7.10	Zusammenfassende Objekte	125
7.11	Darstellung von Hierarchien	127
7.12	Verwaltung von Modellinformationen	130
7.12.1	Das Modellrepository	130
7.12.2	Import von Modellinformationen	130
7.12.3	Codegenerierung	131
7.13	Anbindung von Nutzobjekten	133
	Projekte	133
	Diagramme und Nutzinformationen	133
7.14	Datenbankanbindung	134
	Verwendung von C++-Graphikbibliotheken	134
	Persistenz für graphische Elemente	134
	Entkopplung von Graphikbibliotheken und persistente	
	Diagramminformation	135
7.15	Undo-Mechanismen	136
7.16	Änderbarkeit des generierten Editors	138

8 Zustandsvisualisierung 141

- 8.1 Objektzustände 141
- 8.2 Graphische Darstellung von Objektzuständen 143
- 8.3 Visualisierung von Zustandsänderungen 145
- 8.4 Nebenläufigkeits- und Synchronisationsaspekte 147
- 8.5 Komplexe Visualisierungsszenarien 148

9 Fehlerdetektion in Diagrammen 151

- 9.1 Semantische Fehlerszenarien 151
- 9.2 Formale Beschreibung von Objektgraphen 152
- 9.3 Beschreibung semantischer Fehlerszenarien 154
- 9.4 Vergleich von Fehlerszenarien und Nutzinformationen 155
- 9.5 Vergleichsobjekte 157
- 9.6 Behandlung von (*:n)-Beziehungen 159
- 9.7 Behandlung disjunkter Subgraphen 161
- 9.8 Visualisierung gefundener Fehlerszenarien 162

10 Reverse Engineering 165

- 10.1 Strategie des Algorithmus 166
- 10.2 Aufbau von Aggregationsclustern 167
- 10.3 Traversierung des Nutzobjektgraphen 169
- 10.4 Platzierung der Aggregationscluster 172
- 10.5 Parameter des Algorithmus 176
- 10.6 Komplexität des Algorithmus 176
- 10.7 Beispielszenario 177

11 Anwendungsbeispiele 179

- 11.1 ER-Modelle 179
 - 11.1.1 Fachlicher Hintergrund 179
 - 11.1.2 OOA-Modell 180
 - 11.1.3 Symbole und Verbindungskonstellationen 180
 - 11.1.4 Anwendung 181
- 11.2 Objektorientierte Systeme 183
 - 11.2.1 Fachlicher Hintergrund 183
 - 11.2.2 OOA-Modell 183
 - 11.2.3 Symbole und Verbindungskonstellationen 184
- 11.3 Lineare Impedanznetzwerke 187
 - 11.3.1 Fachlicher Hintergrund 187
 - 11.3.2 Objektorientiertes Modell 188
 - 11.3.3 Symbole und Verbindungskonstellationen 189
 - 11.3.4 Anwendung 190
- 11.4 Logische Schaltwerke 192
 - 11.4.1 Fachlicher Hintergrund 192
 - 11.4.2 OOA-Modell 193
 - 11.4.3 Symbole und Verbindungskonstellationen 194
 - 11.4.4 Anwendung 196
- 11.5 Netzpläne für Projektplanungssysteme 197
 - 11.5.1 Fachlicher Hintergrund. 197
 - 11.5.2 OOA-Modell 198
 - 11.5.3 Symbole und Verbindungskonstellationen 199
 - 11.5.4 Anwendung 200

12 Zusammenfassung und Ausblick 203

- 12.1 Zielsetzung und Lösungsansatz 203
- 12.2 Stand der Entwicklung 205
- 12.3 Erfahrungen und Anwendungsgebiete 206

- 12.4 Weitere Arbeiten 207
- 12.4.1 Erweiterte Graphikfunktionalität 207
- 12.4.2 Einbeziehung von Verbindungskonstellationen in die Zustandsvisualisierung 207
- 12.4.3 Unterstützung hierarchischer Strukturen 207
- 12.5 Portierung auf Java 208
- 12.5.1 Integration von Generatorsystem und Diagrammeditoren 209
- 12.5.2 Laufzeittypenzugriff statt Modellrepository 209
- 12.5.3 Generische Modellierung von Symbolen und Verbindungskonstellationen 211
- 12.5.4 Zustandsvisualisierung und Prädikatfilter 212

Literatur 215

Nachwort 225

A OOA-Modelle der Beispielanwendungen 227

- A.1 ER-Modelle 227
- A.2 OOA-Modelle 229
- A.3 Lineare Impedanznetzwerke 231
- A.4 Logische Schaltwerke 233
- A.5 Projektplanungssysteme 237

B Implementierungsaspekte 241

- B.1 Softwaremetriken 241
- B.2 OOA-Modelle 243
- B.2.1 Metamodell des Repositorys 243
- B.2.2 Modell des Symboleditors und des Editors für

	Verbindungskonstellationen	247
B.2.3	Framework für das Editor-Framework	249
B.3	Operationenprotokolle	250
B.3.1	Glyph	251
B.3.2	InputHandler	251
B.3.3	cneConnType	252
B.3.4	cdeGeoSym	253
B.3.5	cdeBaseSym	253
B.3.6	cdeConn	254

Glossar 255

Lebenslauf 259

Sachindex 261

Namensindex 267