

Inhaltsverzeichnis

1	Einführung	1
1.1	Grundlegende Begriffe	1
1.1.1	Echtzeitbetrieb	1
1.1.2	Echtzeitanforderungen	2
1.1.3	Nebenläufige Rechenprozesse	5
1.2	„Echtzeitdenke“: Kategorien und Optimalität	7
2	Entwurf und Analyse verteilter Echtzeitsysteme	9
2.1	Einleitung	9
2.2	Spezifikations-PEARL	10
2.2.1	Modellierung von Software/Hardware-Architekturen	12
2.2.2	Task-Modellierung	12
2.2.3	Konfigurationsverwalter und Betriebssystem	20
2.3	Simulative Verifikation von S-PEARL-Systemmodellen	23
2.3.1	Systemmodell	24
2.3.2	Verifikation zeitgerechter Verarbeitbarkeit	26
2.4	UML-Profil für S-PEARL	28
2.4.1	Abbildung von S-PEARL auf ein UML-Profil	30
2.4.2	UML-Applikationsarchitektur mit S-PEARL-Stereotypen	45
2.5	Beispiel: Modellierung eines verteilten Automobilbordsystems	45
3	Synchronisation und Konsistenz in Echtzeitsystemen	49
3.1	Einführung	49
3.1.1	Kritischer Bereich	49
3.1.2	Problem des gegenseitigen Ausschlusses	50
3.1.3	Kooperationsproblem	50
3.1.4	Erste Lösungsideen	50
3.2	Semaphore	51
3.2.1	Idee	51
3.2.2	Umsetzung	51

3.2.3	Beispiele	54
3.2.4	Verklemmungen	56
3.3	Monitore	57
3.3.1	Idee	57
3.3.2	Umsetzung	58
3.3.3	Beispiele	59
3.3.4	Mögliche Problemsituationen	62
3.4	Bedingungsgesteuerte Synchronisation	63
4	Echtzeitbetriebssysteme	67
4.1	Einführung	67
4.2	Echtzeittypische Merkmale von Echtzeitbetriebssystemen	68
4.2.1	Prozessorverwaltung	68
4.2.2	Speicherverwaltung	69
4.2.3	Geräteverwaltung	70
4.2.4	Schichtenaufbau	70
4.2.5	Beispiel: Das Betriebssystem RTOS-UH	70
4.3	Prozesse	71
4.3.1	Umsetzung des Mehrprozessbetriebs	74
4.3.2	Problemsituationen	78
4.4	Unterbrechungsbehandlung	79
4.4.1	Beispiele und Problemsituationen	83
4.5	Prozessorzuteilung	84
4.5.1	Umsetzung	84
4.5.2	Beispiele und Problemsituationen	90
5	Echtzeitkommunikation	97
5.1	Echtzeitanforderungen an Kommunikationssysteme	97
5.1.1	Charakteristik von Echtzeitanwendungen	97
5.1.2	Echtzeitanforderungen an die Übertragung von Multimediadaten	98
5.1.3	Echtzeitanforderungen in der industriellen Kommunikation	98
5.2	Architektur von Echtzeitkommunikationssystemen	101
5.3	Transaktionsmodelle	103
5.3.1	Client/Server-Modell	103
5.3.2	Erzeuger/Verbraucher-Modell	104
5.4	Sicherungsschicht	105
5.4.1	Medienzugriffssteuerung	106
5.4.2	Fehlersicherung	111
5.5	Uhrensynchronisation	118
5.5.1	Grundlagen	118
5.5.2	Verfahren zur Uhrensynchronisation	122

6	Programmierung	129
6.1	PEARL	129
6.1.1	Kurzeinführung	129
6.1.2	Beispiele typischer Konstrukte	130
6.1.3	Der BOLT-Typ	131
6.1.4	Einplanungen	132
6.1.5	Zugriff auf Prozessperipherie	137
6.2	Ada	137
6.2.1	Kurzeinführung	137
6.2.2	Beispiel Erzeuger-Verbraucher-Problem	138
6.2.3	Module und Generizität	140
6.2.4	Typen und Anweisungen	142
6.2.5	Tasks und Zeiteinplanungen	144
6.2.6	Ravenscar-Profil für hoch zuverlässige Echtzeitsysteme	147
6.2.7	Zugriff auf Prozessperipherie	150
6.3	Real-Time Specification for Java (RTSJ)	155
6.3.1	Warum Java?	155
6.3.2	Kurzeinführung	156
6.3.3	Schwächen von Java SE bzgl. Echtzeitbetrieb	156
6.3.4	API-Erweiterungen durch RTSJ	157
6.3.5	Eigenschaften der RTSJ	157
6.3.6	Implementierungen der RTSJ	160
6.3.7	Beispiele	161
6.3.8	Synchronisation	167
6.3.9	Probleme	170
7	Qualitätssicherung von Echtzeitsystemen	171
7.1	Qualitätsgerichteter Software-Entwurf	171
7.1.1	Fehlervermeidung	172
7.1.2	Fehlertoleranz	174
7.2	Qualitätssicherung von Software	177
7.2.1	Maßnahmen zur Software-Qualitätssicherung	177
7.2.2	Planung der Software-Qualitätssicherung	178
7.2.3	Struktur von Entwicklungsprojekten	181
7.2.4	Software-Anforderungsspezifikation	182
7.3	Prinzipien von Programmentwurf und -codierung	182
7.4	Software-Diversität	184
7.4.1	Vollständige Diversität	185
7.4.2	Gezielte Diversität	186
7.4.3	Übersetzerdiversität	187
7.4.4	Diversitäre Implementierung	190
7.4.5	Diversitäre Spezifikation	190
7.4.6	Funktionelle Diversität	191
7.4.7	Zur Anwendung der Diversitätsarten	192
7.4.8	Mehrkanalige Software-Realisierung	192

7.5	Richtlinien zur Software-Erstellung für eingebettete Systeme ..	192
7.5.1	Details von Anforderungsspezifikationen	193
7.5.2	Entwurfsprozeduren	196
7.5.3	Software-Struktur	198
7.5.4	Selbstüberwachung	199
7.5.5	Entwurf und Codierung im Detail	200
7.5.6	Sprachabhängige Empfehlungen	202
7.5.7	Sprachen und Übersetzer	203
7.5.8	Systematische Testmethoden	204
7.5.9	Hardware-Erwägungen	205
7.6	Qualitätssicherung von Dokumentationen	206
7.7	Qualitätssicherung von Programmen	207
7.7.1	Verifikationsplan	209
7.7.2	Verifikationstechniken	210
7.7.3	Anforderungsverifikation	212
7.7.4	Entwurfsverifikation	212
7.7.5	Modul- und Codeverifikation	213
7.7.6	Integrationsverifikation von Hard- und Software	213
7.7.7	Rechensystemvalidierung	214
7.8	Verfahren zur Software-Prüfung	215
7.8.1	Inspektionsverfahren	215
7.8.2	Begutachtungen	218
7.8.3	Revisionen	220
7.8.4	Strukturiertes Nachvollziehen	220
7.8.5	Entwurfs- und Codeinspektionen	221
7.8.6	Programmtests	222
7.8.7	Diversitäre Rückwärtsanalyse	227
7.9	Validierung von Echtzeitsystemen	231
7.9.1	Ereignissimulation	231
7.9.2	Externe Umgebungssimulation und Ausgabeverifikation	232
8	Leistungsbewertung und Dienstqualität von	
	Echtzeitsystemen	239
8.1	Leistung von Echtzeitsystemen	239
8.2	Leistungsbewertung von Echtzeitsystemen	241
8.2.1	Beispiele für Benchmark-Programme	242
8.2.2	Laufzeitanalysatoren	243
8.2.3	Leistungsmonitore	245
8.3	Kriterien der Dienstqualität von Echtzeitsystemen	245
8.3.1	Vorhersehbarkeit und Verlässlichkeit	245
8.3.2	Qualitativ-exklusive Kriterien	246
8.3.3	Qualitativ-graduelle Kriterien	248
8.3.4	Quantitative Kriterien	251
8.4	Schlussbemerkung	253