

Inhaltsverzeichnis

1. Einleitung	1
1.1 Problemstellung und Lösungsansatz	1
1.2 Zielsetzung und Abgrenzung des Themas	6
1.3 Thesen	7
2. Bisherige Entwicklungen auf dem Gebiet der Programmkorrektheitsbeweisführung	8
2.1 Historische Übersicht	8
2.1.1 Der Ansatz von Floyd zur Programmkorrektheitsbeweisführung	8
2.1.2 Die zunehmende Problematik der Softwareentwicklung	9
2.1.3 Die Korrektheitsansätze von Hoare, Dijkstra und Mills	9
2.1.4 Die Entwicklung von Spezifikationsprachen	10
2.1.5 Grundsätzliche Fragen zur Programmkorrektheitsbeweisführung	12
2.1.6 Anwendungsorientierung der Programmkorrektheitsfachliteratur	12
2.1.7 Beweisführungswerkzeuge	13
2.1.8 Operationale, denotationale und axiomatische Programmsemantik	14
2.1.9 Alternativen zur Softwarequalitätssicherung	14
2.1.10 Grenzgebiete der Programmkorrektheitsbeweisführung	16
2.1.11 Korrektheitsbeweisführung und objektorientierte Programmierung	17
2.1.12 Programmkorrektheitsbeweise und Normen	17
2.1.13 Gegenwärtige Forschungsprojekte	17
2.1.14 Korrektheitsbeweisführungstechniken im praktischen Einsatz	18
2.1.15 Schlußbemerkungen	19
2.2 Gegenwärtige Situation	20
2.2.1 Der Gegensatz zwischen Theorie und Anwendungspraxis	20
2.2.2 Anforderungen an einen praxisgerechten Ansatz zur Programmkorrektheitsbeweisführung	23
3. Eine praxisgerechte theoretische Grundlage für die Programmkorrektheitsbeweisführung	28
3.1 Grundbegriffe, -betrachtungen und Definitionen	31
3.1.1 Programmvariablen, Datenumgebungen und Ausführungsgeschichten	31
3.1.2 Werte von Variablen und Ausdrücken in Datenumgebungen	32
3.1.3 Programmanweisungen als Funktionen auf \mathbb{D}	32
3.1.4 Programmanweisungen als Funktionen auf \mathbb{D}^*	35
3.1.5 Vorbedingungen und Nachbedingungen	35
3.1.6 Nachbedingungen mit Bezug auf vorherige Variablenwerte	38

3.2 Allgemein gültige Sätze ("Beweisregeln")	39
3.2.1 Stärkung einer Vorbedingung, Schwächung einer Nachbedingung . . .	39
3.2.2 Programmanweisungen und ihre Zusammensetzungen	39
3.2.3 Zerlegung von Vor- und Nachbedingungen	41
3.2.4 Programmsegment, Unterprogramm	41
3.2.5 Beweisregeln für strikte, halbstrikte und umfassende Vorbedingungen	42
3.3 Korrektheitsbeweissführung für sequentielle Programme	44
3.3.1 Voraussetzungen	44
3.3.2 Vorgehensweise ohne maschinelle Unterstützung	46
3.3.3 Anwendungsbeispiele	48
3.3.4 Potentielle Fallen	55
3.3.4.1 Herausnehmen eines Terms aus einer Reihe	55
3.3.4.2 Zuweisung zu einer Feldvariable	56
3.3.4.3 Nicht definierte Ausdrücke	59
3.3.4.4 Unterprogrammaufruf mit formaler Parameterübergabe	60
3.3.4.5 Computerarithmetik	61
3.3.4.6 Rekursion	62
3.4 Mathematische Anforderungen an den Softwareentwickler	63
3.4.1 Vorkenntnisse	64
3.4.2 Verwendete Notationsformen	65
3.5 Stufenweise Gestaltungsmöglichkeiten für unterschiedliche Anwenderkate-	
gorien	66
3.5.1 Mathematische Vorkenntnisse, Terminologie und Schreibweise	67
3.5.2 Informelle gegenüber formelle Anwendung	68
3.5.3 Abstraktionsgrad der Spezifikation und Aufgabenstellung	71
4. Bedeutung der Korrektheitsbeweissführung für die ingenieurmäßige Neukon-	
 struktion eines Programms	74
4.1 Aus der Korrektheitsbeweissführung sich ergebende Anforderungen an und	
Leitlinien für die Konstruktion	75
4.2 Ansätze und Möglichkeiten zur Ableitung von Teilen eines Programms . . .	79
4.3 Konstruktionsbeispiel	81
4.4 Spezifikationen bei der Konstruktion und im Korrektheitsbeweis	85
4.5 Anforderungen an die Dokumentation eines Unterprogramms	88
5. Bedeutung der Korrektheitsbeweissführung für die Konstruktionsänderung	
 in instabiler Umgebung	91
5.1 Spezifikationen von Unterprogrammen	92
5.2 Eingrenzung der Auswirkungen einer Änderung	95
5.3 Konstruktionsleitlinien für die "Änderungsfreundlichkeit"	100

6. Anwendungsaufwand, Voraussetzungen und maschinelle Unterstützung für eine breitere Anwendungsakzeptanz	102
6.1 Lernphase	103
6.2 Programmverifikation	104
6.2.1 Manuelle Korrektheitsbeweissführung	105
6.2.2 Maschinelle Unterstützung	108
6.3 Programmkonstruktion	110
6.4 Konstruktionsänderung	111
7. Spezielle Aspekte der Programmkorrektheitsbeweissführung	112
7.1 Computerarithmetik	112
7.1.1 Ganzzahlenarithmetik auf einem beschränkten Intervall	112
7.1.2 Gleitkommaarithmetik	115
7.2 Nebenläufigkeit aus der Sicht der Korrektheitsbeweissführung	120
7.2.1 Wechselwirkung zwischen nebenläufigen Prozessen in einem Korrektheitsbeweis	121
7.2.2 Ein Korrektheitsbeweis für Kommunikation zwischen nebenläufigen Prozessen	123
7.2.2.1 Übertragungskanal ohne Puffer	123
7.2.2.2 Gepufferter Übertragungskanal	133
7.2.3 Vollständige Korrektheit und Verklemmung/Verhungern (lassen)	136
7.3 Objektorientierte Programmierung am Beispiel der OOP-Sprache Eiffel	137
7.3.1 Korrektheitsbeweissführungsrelevante Besonderheiten von Eiffel	138
7.3.1.1 Variablen und Objekte	138
7.3.1.2 Routinen	140
7.3.1.3 Zugriffseinschränkungen	141
7.3.1.4 Ausnahmebehandlung	142
7.3.1.5 Unbestimmte Bezüge auf Variablen und Funktionen	142
7.3.2 Die korrektheitsbezogenen Konstrukte in Eiffel	143
7.3.3 Beispiel der Korrektheitsbeweissführung für eine Klasse	145
7.3.4 Verbesserungs- und Erweiterungsvorschläge zu den korrektheitsbezogenen Konstrukten in Eiffel	152
8. Schlußfolgerungen	156
8.1 Erreichtes	156
8.2 Als offen Erkanntes	156
8.3 Handlungsbedarf	157

Anhang 1. Beispiele unterschiedlicher Spezifikationsstufen	159
A1.1 Kernreaktorüberwachung	159
A1.1.1 Allgemeine Beschreibung der Überwachungsaufgabe	159
A1.1.2 VDM-Spezifikationen unterschiedlicher Abstraktionsgrade	160
A1.1.3 Anwenderorientierte Vorgehensweise zur Erarbeitung der Spezifikation	162
A1.2 Kellerspeicher	165
A1.2.1 Abstrakte Spezifikationen eines Kellerspeichers (Funktionensystem)	165
A1.2.2 Abstrakte Spezifikation eines Systems parameterloser Unterprogramme	165
A1.2.3 Spezifikation durch Vor- und Nachbedingungen	167
A1.2.4 Konkrete Spezifikationsstufe 1 eines Unterprogrammsystems	168
A1.2.5 Konkrete Spezifikationsstufe 2 eines Unterprogrammsystems	168
Anhang 2. Dokumentationsbeispiel mit Korrektheitsbeweis für den Programmteil "Aufteilen"	171
Anhang 3. Einige Fehlerschranken für Gleitkommaarithmetik	186
A3.1 Definition und Axiome eines Gleitkommaarithmetiksystems	186
A3.2 Genauigkeit der Gleitkommadarstellung	188
A3.3 Genauigkeit der Gleitkommaaddition	189
A3.4 Genauigkeit der Gleitkommamultiplikation	190
A3.5 Rundungseigenarten	190
Anhang 4. Korrektheit eines rekursiven Unterprogramms	191
Literatur- und Quellenhinweise	201
Literaturhinweise	201
Persönliche Kommunikation	236