

Inhaltsverzeichnis

Abbildungsverzeichnis	vii
Tabellenverzeichnis	ix
Listingsverzeichnis	xi
1 Einführung	1
1.1 Zielsetzung	3
1.2 Aufbau der Arbeit	5
2 Java	7
2.1 Die Java-Technik	8
2.2 Die Programmiersprache Java	9
2.2.1 Historie und Entwurfsziele	9
2.2.2 Grundlegende Sprachmerkmale	11
2.3 Die Java Virtual Machine	13
2.3.1 Architektur	14
2.3.2 Speicherarchitektur	16
2.3.2.1 Methodenbereich	16
2.3.2.2 Heap	17
2.3.2.3 Thread-Speicher	17
2.3.3 Ausführungseinheit	22
2.3.4 Datentypen	24
2.3.5 Bytecode-Instruktionen	25
2.3.5.1 Aufbau	25
2.3.5.2 Datenflussoperationen	26
2.3.5.3 Kontrollflussoperationen	29
2.3.5.4 Rechenoperationen	31
2.3.6 Format der Java-Klassendatei	32
2.4 Fazit	34

3	Grundlagen der Fehlersuche	37
3.1	Ursache und Wirkung von Programmfehlern	37
3.2	Infektionstypen	40
3.3	Debugging von Programmen	45
3.4	Fazit	48
4	Debugging-Techniken	49
4.1	Trace-Debugging	50
4.2	Omniscient-Debugging	54
4.3	Deklaratives Debugging	57
4.3.1	Verfahren	57
4.3.2	Adaption für die Programmiersprache Java	66
4.3.2.1	Berechnungsbaum für Java-Programme	66
4.3.2.2	Seiteneffekte in Java-Programmen	66
4.3.2.3	Erforderliche Informationen zur Klassifizierung eines Methodenaufrufs	67
4.3.2.4	Der Zustandsraum eines Methodenaufrufs	72
4.3.2.5	Aufwandsvergleich zu deklarativen Sprachen	74
4.4	Hybrides Debugging	76
4.5	Fazit	80
5	Deklarative Debugging-Strategien	83
5.1	Top-Down	85
5.2	Divide-and-Query	86
5.3	D&Q mit gewichtsunabhängigen Infektionswahrscheinlichkeiten	91
5.3.1	Verfahren	91
5.3.2	Kontrollflussgraph	95
5.3.3	Kontrollflussbasierte Infektionswahrscheinlichkeit	95
5.3.4	Datenflussorientierte Infektionswahrscheinlichkeit	100
5.3.5	Schätzung der Infektionswahrscheinlichkeit	103
5.4	Empirische Untersuchung	107
5.5	Fazit	111
6	Der Java-Hybrid-Debugger	113
6.1	Benutzeroberfläche	113
6.1.1	Berechnungsbaumansicht	115
6.1.2	Knotenansicht	117

6.1.3	Ereignisansicht	121
6.1.4	Variablenansicht	126
6.2	Debugging eines defekten Java-Programms	128
6.2.1	Defekter Mergesort-Algorithmus	128
6.2.2	Suche des Defekts mit JHyde	131
6.3	Fazit	144
7	Entwurf und Implementierung	145
7.1	Architektur	146
7.2	Transmitter	148
7.2.1	Die Ereignisschnittstelle	149
7.2.1.1	Klassenstruktur	150
7.2.1.2	Erzeugen von Objekten	152
7.2.1.3	Lesen von Variablen	153
7.2.1.4	Schreiben von Variablen	154
7.2.1.5	Methodenaufruf	154
7.2.1.6	Sonstiger Kontrollfluss	155
7.2.2	Architektur	156
7.2.3	Sendeprozess	159
7.2.4	Empfangsprozess	164
7.2.5	Konfiguration	168
7.3	Instrumentierer	169
7.3.1	Instrumentierung von Java-Programmen	169
7.3.1.1	Java-Instrumentation-API	170
7.3.1.2	Frameworks	171
7.3.2	Klassen in der Prüflings-VM	176
7.3.3	Instrumentierungsschema	178
7.3.4	Architektur	193
7.3.5	Instrumentierungsprozess	194
7.3.5.1	Komposition der Visitor-Instanzen	195
7.3.5.2	Dynamische Instrumentierung	196
7.3.5.3	Statische Instrumentierung	199
7.3.5.4	Instrumentierung der Registrierungsmecha- nismen	199
7.3.6	Konfiguration	200
7.4	Rekorder	201
7.4.1	Modell des Programmablaufs	201

7.4.2	Debugging-Strategien	205
7.5	Benutzeroberfläche	209
7.6	Fazit	213
8	Stand der Forschung	217
8.1	Abfragebasiertes Debugging	218
8.2	Record-Replay-Techniken	221
8.3	Omniscient-Debugging	227
8.4	Deklaratives Debugging	229
8.5	Debugging-Strategien	232
8.6	Fazit	235
9	Schlussbetrachtungen	237
9.1	Zusammenfassung	237
9.2	Ausblick	240
	Literaturverzeichnis	243