Katharina Mehner

Trace-based Debugging and Visualisation of Concurrent Java Programs with UML

Zusammenfassung

Nebenläufige Programe sind aufgrund des inhärenten Nichtdeterminismus besonders anfällig für Fehler, die von der Interaktion nebenläufiger Kontrollflüsse herrühren. Entwickler nebenläufiger Java-Anwendungen erhalten jedoch keine gezielte Unterstützung bei der Suche nach Nebenläufigkeitsfehlern. Weder sind diese Fehler ausreichend dokumentiert, noch ist die Unterstützung der Fehlersuche zur Laufzeit durch Werkzeuge wie Debugger zufriedenstellend.

Diese Arbeit präsentiert einen Ansatz zur automatischen Erkennung von Lebendigkeitsfehlern, einer besonderen Klasse von Nebenläufigkeitsfehlern, während der Ausführung von Java-Anwendungen. Dazu werden Lebendigkeitsfehler und entsprechende Potentiale analysiert und klassifiziert. Zur Beschreibung des Verhaltens und der Interaktion von nebenläufigen Kontrollflüssen wird ein UML-Statechart entwickelt. Basierend auf dessen Zuständen und Ereignissen werden Lebendigkeitsfehler und ihre Potentiale formal spezifiziert.

Ausgehend von diesen Spezifikationen werden Algorithmen zur Fehlererkennung entwickelt. Zur Visualisierung von Programmabläufen und von Fehlern wird ein UML-Profil basierend auf Interaktionsdiagrammen definiert. Zur Umsetzung dieser Konzepte wird zum einen ein Datenformat für Traces, d.h. für die Protokollierung von Programmabläufen, definiert. Zum anderen wird eine Methode zur Trace-Erzeugung sowie eine Methode zur Visualisierung beschrieben. Beide Methoden sind in der prototypischen Umgebung JAVIS umgesetzt. Diese besteht aus einem Java-Tracer mit einer Analysefunktionalität zur Überwachung der Lebendigkeit nebenläufiger Java-Programme und aus einer Erweiterung für das UML-Werkzeug Together, mit der Java-Traces importiert und visualisiert werden können. In den Traces enthaltene Lebendigkeitsfehler und Potentiale werden in eigenen Diagrammen visualisiert und ihre Ursachen identifiziert.