Modularer Java Profiler

Studiengang: MAS Information Technology
Betreuer: Daniel Tschan, Puzzle ITC GmbH
Experte: Dr. Stephan Fischli, (Berner Fachhochschule)
Industriepartner: Puzzle ITC GmbH, Bern

Ein Profiler ist ein Entwicklungswerkzeug, welches das Laufzeitverhalten einer Applikation aufzeigt. «Modjprof» ist ein modularer Open Source Java Profiler, welcher das Messen von Durchlaufzeiten einzelner Programmteile ermöglicht. Die Messdaten werden für eine effiziente Eingrenzung von Performance-Problemen aufbereitet. Dabei werden einzelne Programmabschnitte zur Laufzeit mit Analysecode erweitert, ohne das Laufzeitverhalten der Applikation nennenswert zu beeinflussen.

Ausgangslage

Puzzle ITC entwickelt stabile und massgeschneiderte Software für diverse Kunden und legt dabei grossen Wert auf qualitativ hochstehende Lösungen. Ein grosser Qualitätsindikator ist dabei die Performance einer Software. Manchmal treten Performance-Probleme bereits während der Entwicklungsphase auf, manchmal zeigen sie sich erst in einer produktiven Umgebung. Die Ursachen dafür können auf verschiedenen Architekturschichten liegen: Beim Datenbankzugriff, bei der Verarbeitung der Daten, in der Business-Logik oder bei der Darstellung für den Benutzer. Weiter kann das Problem auch in einer fremden Bibliothek liegen.

Um allfällige Performance-Probleme aufzuspüren kann der Entwickler verschiedene Werkzeuge wie Logger, Debugger oder Profiler einsetzen. Letzterer eignet sich oft am besten, da mit dem Profiler das Laufzeitverhalten der Software gezielt in der problematischen Umgebung, zur Laufzeit und ohne Programmanpassungen analysiert werden kann.

Es gibt verschiedene Typen von Profilern. Eine mögliche Technik einen Profiler zu implementieren ist nach dem Prinzip der Instrumentierung. Dabei werden einzelne Programmabschnitte zur Laufzeit mit Analysecode erweitert.

Lösung

Der Profiler ist aus mehreren Komponenten aufgebaut. Die Hauptkomponente ist der Java Agent, welcher für die Instrumentierung der Applikation verantwortlich ist. Durch Konfiguration lässt sich bestimmen welche Applikationsteile und Fremdbibliotheken analysiert werden sollen und welche nicht. Es wird jeweils der Eintritt sowie der Austritt einer Methode durch Analyse-Code ergänzt, damit eine Aussage über die Durchlaufzeit jeder Methode gemacht werden kann. Dabei wird ein Callback auf den Agent selbst eingefügt. Der Agent ist danach für das Speichern der Analysedaten verantwortlich. Das für die Instrumentierung eingesetzte Framework wurde anfänglich evaluiert. Mit Hilfe von Prototypen wurde die Auswirkungen auf die Applikation gemessen und das Framework mit minimalster Beeinflussung des Laufzeitverhaltens ausgewählt. Es ist möglich, dass dasselbe Framework in einer anderen Version bereits in der Applikation verwendet wird. Damit es zu keinen Versionskonflikten mit der Applikation kommt, wird das Framework im Agent mit einem eigenen Classloader geladen und so von der Applikation isoliert.

Ein Control Servlet ermöglicht es, den Profiler auch auf einem entfernten System ohne direkten Zugriff einzusetzen. Somit kann der Agent auch über längere Zeit gestartet und bei Bedarf wieder gestoppt werden. Ausserdem kann von einem entfernten System Zugriff auf die Analysedaten bereitgestellt werden.

Um spezifische Anwendungsfälle einer Webapplikation zu analysieren, wurde ein Request Filter implementiert, mit dem der Agent bewusst für einzelne Browser Requests gestartet werden kann. Somit kann die Auswertung auf einen einzigen Request eingeschränkt werden, um die Datenmenge so gering wie möglich zu halten. Dies vereinfacht es erheblich Performanceprobleme einzugrenzen.

Der Agent schreibt die Analysedaten zur Laufzeit möglichst effizient. Daher müssen diese Daten nachträglich von einer Auswertungskomponente für die Analyse aufbereitet und in ein gut lesbares Format gebracht werden. Die Durchlaufzeiten der Methodenaufrufe werden berechnet, gruppiert und hierarchisch nach Zeitverbrauch sortiert.

Fazit

Der Profiler modjprof ist ein stabiles Entwicklungswerkzeug für die schnelle Eingrenzung von Performance-Problemen. Darüber hinaus kann der Java Agent mit minimaler Beeinflussung des Laufzeitverhaltens in einer produktiven Umgebung eingesetzt werden, um im Bedarfsfall eingeschaltet zu werden und aufschlussreiche Resultate zu liefern. Alles in allem ein Tool, das in keinem Enterprise Java Projekt fehlen sollte.

Philipp Grogg