Available online at www.sciencedirect.com
ScienceDirect
Electronic Notes in Theoretical Computer Science
Electronic Notes in Theoretical Computer Science 264 (4) (2011) 51–57
www.elsevier.com/locate/entcs

JRebel Tool Demo

Jevgeni Kabanov[1]

Dept. of Computer Science
University of Tartu
Tartu, Estonia

Abstract

JRebel started as an academical project that became a successful commercial product used by thousands of developers worldwide. It extends the Java Virtual Machine with a mechanism that allows seamless class reloading. It uses bytecode manipulation extensively, both for the just-in-time class translator and numerous integrations with the Java SE and EE APIs. In this live demo we will show how it can be used in real-life projects to cut development time by 8 to 18 per cent.

Keywords: bytecode, JRebel, ClassLoader, API, retroactive

1 Introduction

Java EE development day-to-day activities involves deploying the application to the Java EE server. This step is necessary after the application has been compiled and packaged into an archive as per Java EE specification. Every time developers want to make changes to the running application they need to deploy it, which can take from a few seconds in the best case to several minutes in the worst.

An alternative way to update an application is using the HotSwap protocol [1], available from the Java EE debugger. This allows to update the application classes without redeploying it. Unfortunately only a very restricted set of changes is allowed; namely HotSwap allows changes to the method bodies, but does not allow changing the class signature or inheritance hierarchy. Thus no new methods, fields or constructors are allowed.

At the end of 2006 we came up with an idea for extending the Java virtual machine with a mechanism that would allow to change the class bytecode beyond the limits of the HotSwap protocol [2]. During 2007 we developed and released a prototype initially code-named “Badger” and for the public release renamed to

[1] Email: ekabanov@gmail.com

doi:10.1016/j.entcs.2011.02.005

"JavaRebel". In 2009 we released the version 2.0, which supported a layer of indirection on top of the ClassLoader API that allowed the users to edit classes and resources in their workspace instead of packaging them into .WAR or .EAR archives as per Java EE specification. We also introduced an extension API that allowed to make use of JavaRebel features in third-party applications as well as build plugins for JavaRebel to support changes in framework configuration. We also renamed the tool once more to "JRebel" due to trademark issues.

When we started working on the tool most of the research was focused on using ClassLoaders to dynamically update code [3] or on modifying JVMs to do so [4]. Recently there has been more investigation into similar systems [5,6], but no industry tools are available to compete with JRebel.

At the moment we estimate over ten thousand of JRebel users around the world. This number includes commercial users as well as various otherwise licensed users, e.g. Open Source developers and Scala developers who can request a free license.

The rest of the paper is organized as follows. In Section 2 we review the problem and its scope, Section 3 gives a brief overview of the tool, Section 4 describes the technical background and Section 5 covers the supporting artifacts and third-party extensions.

This work was partially supported by the OÜ Tarkvara Tehnoloogia Arenduskeskus, Enterprise Estonia and Estonian Science Foundation grant No. 8421.

2 Problem Scope

In 2009 we conducted a survey reaching over 1000 Java developers to estimate the amount of time spent in the redeploy phase of development. The survey asked how long a server redeploy takes and how many times an hour they are performed and is summarized by the following chart: