A Small Extension to Java for Class Refinement

Muga Nishizawa
Dept. of Mathematical and Computing Sciences
Tokyo Institute of Technology
2-12-1 Ohkayama, Meguro-ku
Tokyo 152-8552, Japan
muga@csg.is.titech.ac.jp

Shigeru Chiba
Dept. of Mathematical and Computing Sciences
Tokyo Institute of Technology
2-12-1 Ohkayama, Meguro-ku
Tokyo 152-8552, Japan
chiba@is.titech.ac.jp

ABSTRACT

This paper presents an extended Java language in which users can refine a class definition to a certain degree. They can statically or dynamically redefine methods and append a new method, field, and interfaces to the class like dynamic languages. A unique feature of this language, named *GluonJ*, is that users can use a standard Java IDE (Integrated Development Environment) to exploit coding support by the IDE. This is significant for the industrial acceptability of a new language. A GluonJ program is written in standard Java with additional Java annotations. GluonJ was carefully designed so that the IDE can recognize a GluonJ program and reflect it on the coding support such as the code assist of Eclipse. Moreover, a GluonJ program never throws a runtime exception reporting that an undefined method is called. Guaranteeing this property is not straightforward because GluonJ allows users to refine a class definition at runtime.

Categories and Subject Descriptors

D.3.3 [**Programming Languages**]: Language Constructs and Features

General Terms

Languages, Design

Keywords

Java, Class refinement, Programming transformation

1. INTRODUCTION

Software evolution is one of the most significant topics in the software industry. To react to altering and evolving requirements at a rapid pace, software must be extended quickly. To minimize this effort, the extensions should be implemented in a modular fashion as much as possible.

Refinement is one of the promising technologies for this. The concept of refinement is similar to mixin layers, virtual classes, aspect-oriented programming (AOP), feature-oriented programming (FOP), and so on. It is a language mechanism for extending an existing class[1]. Unlike subclassing and mixin mechanisms, it directly modifies the definition of an existing class. Thus, a client program can use a method overridden by refinement without explicitly creating an instance of the extended version of that class.

This paper proposes our extended Java language, named *GluonJ*, in which users can statically or dynamically refine the definition of an existing class. Our contribution is the pragmatic design of GluonJ's language construct for refinement. Our extension to Java is small. GluonJ uses Java annotations and thereby does not extend the lexical syntax of Java. It exploits Java's type system as much as possible. Thanks to these, a GluonJ program can be developed on a standard Java IDE (Integrated Development Environment) such as Eclipse and NetBeans. Particularly, users can enjoy the coding supports by a standard IDE even for GluonJ programming. The IDE can recognize methods appended by refinement and shows them as candidates when its users are typing a method name. A GluonJ program is compiled by a standard Java compiler. Only a special runtime system is needed to run a GluonJ program. We introduced these features for industrial acceptability.

Although GluonJ enables appending/removing a method to/from an existing class according to dynamic contexts, a GluonJ program never throws a NoSuchMethodException if it is successfully compiled and loaded. A naive implementation of dynamic-refinement might wrongly allow a client to call an unavailable method, that would be appended later by refinement but that does not yet exist. To avoid such an invalid call, which would throw a runtime exception, GluonJ requires users to follow some programming conventions. A GluonJ program satisfying these conventions never calls an unavailable method. If a program does not satisfy the conventions, they are statically detected before the program starts running. To do this, GluonJ exploits Java's type system and GluonJ's custom class loader.

2. GLUONJ

This paper proposes our extended Java language, named *GluonJ*, in which users can statically or dynamically refine the definition of an existing class. The users of GluonJ can redefine existing methods and append new methods, fields, and interfaces to an existing class. Since these changes are described in a separate component (or module), this language mechanism is useful for separation of concerns. Refinement is described with annotations within the standard Java syntax. Thus, a GluonJ program can be compiled by a standard Java compiler although compiled bytecode is transformed by GluonJ's custom class loader.

SAC'08 March 16-20, 2008, Fortaleza, Ceará, Brazil
Copyright 2008 ACM 978-1-59593-753-7/08/0003 ...$5.00.

[1] The term ”refinement” is generally used in the context of formal methods. However, in this paper, it is used as a language mechanism for the extension to an existing class.