Master Thesis

Implementing a Universe Type System for Scala

Manfred Stock

August 2007 – January 2008

Introduction The Universe Type System [2] is used to control aliasing and dependencies in object-oriented programs. Its underlying basis is the concept of ownership where each object is owned by at most one owner object. The objects are organized into *contexts*, which are sets of objects with the same owner. Objects without owner are grouped into the so called *root context*, which also forms the root of the tree of contexts of a program execution. When enforcing the so-called owner-as-modifier discipline, the owner must have control over the modification of its objects.

Scala [3] is a programming language combining features of object-oriented and functional programming. It compiles to Java bytecode and is therefore interoperable with Java. Scala also features a powerful annotation system which is far superior to the Java 1.5 annotations.

Goal of this master thesis is the implementation of a Generic Universe Type [1] checker for Scala using its annotation system and pluggable type checker. An earlier master thesis [5] already built a prototype implementation of a Universe Type System for Scala, but could not make use of pluggable types since they were not implemented in Scala, yet. Recent work now separates the Universe Type System from the owner-as-modifier discipline, making the latter an optional add-on to the Universe Type System. This thesis will therefore also be used to evaluate this new variant.

The main parts of this project are:

1. Implementation of a pluggable type checker for the Generic Universe Type System.

2. Implementation of run time checks for Scala like it was done in [4] for Java.

3. Case studies, both small and larger, using the above implementations.

4. Local type inference also for Universe Types: Scala can often infer types of local variables and function return values. This should be implemented for Universe Types as well.

5. Definition and implementation of path-dependent Universe Types.

Possible extensions include the support for first-class functions and encapsulation policies which can be used as alternatives to the owner-as-modifier discpline.

References

[1] W. Dietl, S. Drossopoulou, and P. Müller. Generic Universe Types. In E. Ernst, editor, *European Conference on Object-Oriented Programming (ECOOP)*, Lecture Notes in Computer Science. Springer-Verlag, 2007. To appear.

[2] Werner Dietl and Peter Müller. Universes: Lightweight Ownership for JML. *Journal of Object Technology*, 4(8):5–32, 2005. `http://www.jot.fm/issues/issue_2005_10/article1`.

[3] Martin Odersky et al. Scala. `http://www.scala-lang.org/`.

[4] Daniel Schregenberger. Runtime Checks for the Universe Type System, 2004. Semester Thesis.

[5] Daniel Schregenberger. Universe Type System for Scala. Master's thesis, Swiss Federal Institute of Technology Zurich (ETHZ), Department of Computer Science, 2007.