Expressive Modular Linking for Object-Oriented Languages

UUCS-02-014

Sean McDirmid, Wilson C. Hsieh, Matthew Flatt
School of Computing
University of Utah
{mcdirmid,mflatt,wilson}@cs.utah.edu

October 11, 2002

In this paper we show how modular linking of program fragments can be added to statically typed, object-oriented (OO) languages. Programs are being assembled out of separately developed software components deployed in binary form. Unfortunately, mainstream OO languages (such as Java) still do not provide support for true modular linking. Modular linking means that program fragments can be separately compiled and type checked, and that linking can ensure global program type correctness without analyzing program fragment implementations.

Supporting modular linking in OO languages is complicated by two expressive features that current OO languages do not support together: mixin-style inheritance across program fragment boundaries, and cyclic dependencies between program fragments. In a previous paper at OOPSLA 2001, we have demonstrated the practical uses for such expressiveness. When such expressiveness is permitted, link-time type checking rules must ensure that method collisions and inheritance cycles do not occur after program fragments are linked into a program. In this paper, we show how modular linking with both cyclic linking and mixin-style inheritance can be supported using a type-checking architecture that can be added on top of existing OO languages, such as Java.