ABSTRACT

This dissertation provides a framework for modularity in programming languages. In this framework, known as *Jigsaw*, inheritance is understood to be an essential linguistic mechanism for module manipulation.

In *Jigsaw*, the roles of classes in existing languages are "unbundled," by providing a suite of operators independently controlling such effects as combination, modification, encapsulation, name resolution, and sharing, all on the single notion of *module*.

All module operators are forms of inheritance. Thus, inheritance is not in conflict with modularity in this system, but is indeed its foundation.

This allows a previously unobtainable spectrum of features to be combined in a cohesive manner, including multiple inheritance, mixins, encapsulation and strong typing.

Jigsaw has a rigorous semantics, based upon a denotational model of inheritance.

Jigsaw provides a notion of modularity independent of a particular computational paradigm. *Jigsaw* can therefore be applied to a wide variety of languages, especially special-purpose languages where the effort of designing specific mechanisms for modularity is difficult to justify, but which could still benefit from such mechanisms.

The framework is used to derive an extension of *Modula-3* that supports the new operations. An efficient implementation strategy is developed for this extension. The performance of this scheme is on a par with the methods employed by the highest performance object-oriented language processors currently available.