Abstract

Persistent Application Systems are potentially very large and long-lived application systems which use information technology: computers, communications, networks, software and databases. They are vital to the organisations that depend on them and have to be adaptable to organisational and technological changes and evolvable without serious interruption of service.

Persistent Programming Languages are a promising technology that facilitate the task of incrementally building and maintaining persistent application systems. This thesis identifies a number of technical challenges in making persistent programming languages scalable, with adequate performance and sufficient longevity and in amortising costs by providing general services.

A new architecture to support the compilation of long-lived, large-scale applications is proposed. This architecture comprises an intermediate language to be used by front-ends, high-level and machine independent optimisers, low-level optimisers and code generators of target machine code.

The intermediate target language, TPL, has been designed to allow compiler writers to utilise common technology for several different orthogonally persistent higher-order reflective languages. The goal is to reuse optimisation and code-generation or interpretation technology with a variety of front-ends. A subsidiary goal is to provide an experimental framework for those investigating optimisation and code generation. TPL has a simple, clean type system and will support orthogonally persistent, reflective, higher-order, polymorphic languages. TPL allows code generation and the abstraction over details of the underlying software and hardware layers.

An experiment to build a prototype of the proposed architecture was designed, developed and evaluated. The experimental work includes a language processor and examples of its use are presented in this dissertation. The design space was covered by describing the implications of the goals of supporting the class of languages anticipated while ensuring long-term persistence of data and programs, and sufficient efficiency. For each of the goals, the design decisions were evaluated in face of the results.