発表概要

LMNtal処理系および他言語インタフェースの設計と実装

原 耕司[†] 水野 謙[†] 矢島伸吾[†]
永田貴彦[†] 中島求[†]
加藤紀夫[†] 上田和紀[†]

本発表では，階層的グラフ書換えに基づく言語モデルである LMNtal の処理系開発状況をデモを交えて報告する．また，実用言語に不可欠である入出力などを実現するための仕組みである他言語インタフェースの設計と実装について詳しく説明する．LMNtal は計算に関する多様な考え方を統合することを目標として提案されており，LMNtal の普及のためには実用的な処理系を提供することが重要と考えられる．現在公開されている LMNtal 処理系は約 18,000 行の Java コードから成り，eclipse と CVS を用いて 6 名のメンバーによりチーム開発されている．処理系は，中間命令列へのコンパイラ部分と中間命令列を実行するランタイム部分からなる．実用的なプログラミング言語の処理系にとって入出力など OS の機能を呼び出す仕組みは必要不可欠である．また，プログラマが記述した他言語コードをユーザプログラムから簡単な方法で呼び出したい場合もあろう．我々は，これらの要求を統一的に実現するために必要となる他言語インタフェースを LMNtal 処理系上に設計し，実装した．他言語コードは，アトムとしてソースプログラム中にインライン記述する．処理系のコンパイラ部分は，インラインコードから Java ソースファイルを生成し，クラスファイルにコンパイルして動的に読み込む．インラインコードは，対応するアトムを生成するときにランタイム部分によって呼び出される．

Design and Implementation of the LMNtal System and Its Foreign-Language Interface

Koji Hara,[†] Ken Mizuno,[†] Shingo Yajima,[†] Takahiko Nagata,[†]
Motomu Nakajima,[†] Norio Kato[†] and Kazunori Ueda[†]

We demonstrate the current status of the development of the LMNtal system, the language model based on hierarchical graph rewriting. We also explain in detail the design and implementation of the foreign-language interface needed to perform I/O operations, which is indispensable for a practical language system. LMNtal is a language model proposed for unifying various notions on computaion, and it would be important for that purpose to offer a practical language system. The latest implementation of the LMNtal system has 18,000 lines of Java code and is developed by six members with eclipse and CVS. The system consists of the compiler part for generating intermediate language instructions and the runtime part that executes them. The mechanism to call operating system functions, such as input and output, is indispensable for a practical language system. We may sometimes want to call foreign-language code from a user program in a simple way. We have designed and implemented a foreign-language interface on top of the LMNtal system in order to realize these demands systematically. Foreign-language code is described inline as an atom in the source program. The compiler part of the system generates Java files from inline code, compiles them to classfiles, and loads them dynamically. The runtime part then executes the code when the corresponding atom is generated in a rewriting.

（平成 16 年 7 月 30 日発表）

† 早稲田大学大学院理工学研究科情報・ネットワーク専攻
Department of Computer Science, Waseda University