The Grid-Occam Project

I. Overview

Project Title

The Grid-Occam Project

Contributors Names

Andreas Polze, Peter Tröger, Martin von Löwis

Abstract

Occam is a parallel processing language designed by a team at INMOS in conjunction with the design of the transputer processor, and based on Sir T. Hoare's ideas of Communicating Sequential Processes (CSP). Occam incorporates support for very fine grained, easy to use threads and seamless support of multi-processor environments. It can be used with shared or distributed memory systems, and the strong basis in CSP makes it excellent choice when formal proofs of correctness are required.

Within the proposed project, we plan to develop an implementation of Occam on Rotor as a vehicle for education and teaching. Our work will especially focus on concurrency issues and investigate the suitability of the Occam programming model for highly concurrent and parallel programming on multiprocessor and multicomputer systems.

Extending on our previous Rotor project “Object and Process Migration in Rotor and .NET”, we plan to use the platform independent MSIL-format as a vehicle to bridge machine boundaries. Therefore, the Grid-Occam project will develop a distributed implementation of Occam on Rotor/.NET as well, thus extending the Occam/CSP programming model into the arena of heterogeneous distributed computing.

Our research activities in the project will be accompanied by a series of lectures and students’ projects, which will allow us to teach principles of compiler construction, language processing tools, and concurrent programming based on Rotor.

Institution Details

Prof. Dr. Andreas Polze
Operating Systems and Middleware Group
Hasso-Plattner-Institute for Software Engineering
at University Potsdam
Prof.-Dr.-Helmert Str. 2-3
14482 Potsdam
Germany

www.dcl.hpi.uni-potsdam.de
www.hpi.uni-potsdam.de

Contributor Details

- Prof. Dr. rer.-nat. habil. Andreas Polze
- Head of Operating Systems and Middleware Group (OSM)
- Hasso-Plattner-Institute for Software Engineering at University Potsdam, Germany
- **polze@hpi.uni-potsdam.de**, andreas@polze.de
- +49-331-5509-231 (Secretary: -220)
- andreas.polze.de

- Dr. Martin von Löwis
- Lecturer at the Operating System and Middleware Group
- **martin.vonLoewis@hpi.uni-potsdam.de**

- Dipl.-Inf. Peter Tröger
- Ph.D. student at the Operating System and Middleware Group
- **peter.troeger@hpi.uni-potsdam.de**

Prof. Dr. Andreas Polze received a doctoral degree from Freie University Berlin in 1994 and a habilitation degree from Humboldt University Berlin in 2001, both in Computer Science. His habilitation thesis focuses on *Predictable Computing in Multicomputer-Systems*. In 1997 he spent 7 months as a visiting scientist with the Dynamic Systems Unit at Software Engineering Institute, at Carnegie Mellon University, Pittsburgh, USA, were he worked on real-time computing on standard middleware (CORBA). In Fall 2000 he served as a guest professor at the practical computer science chair at University Osnabrück. Since October 2001, Andreas Polze is Professor at the Hasso-Plattner-Institute for Software Engineering at University Potsdam, Germany, where he heads the Operating Systems and Middleware chair. His current research interests include Interconnecting Middleware and Embedded Systems, Mobility and Adaptive System Configuration, and End-to-End Service Availability for standard middleware platforms. He is member of the GI and the IEEE. He currently is member of the program committees of ISORC (Intl. Symp. On Object-Oriented Real-Time Computing) and WORDS (Workshop on Real-Time Dependable Systems).

II. Project Description

Background

Research at the Operating Systems and Middleware Group at Hasso-Plattner-Institute (HPI) for Software Systems Engineering at University Potsdam is focused on paradigms, design patterns, and software engineering approaches for middleware-based systems. Our work addresses non-functional system behavior and system predictability with respect to real-time, fault-tolerance, and security. We currently investigate online replacement of analytically redundant components, which identifies a new, promising paradigm for dependable and adaptive systems combining COTS-based and embedded components.

We use Rotor as a fascinating infrastructure that allows us to address research questions, such as how to describe a component's resource requirements (CPU, memory, communication bandwidth), which cannot easily be expressed in the component's functional interface. Automatic system configuration based on these component descriptions is the most challenging research question, which is currently under investigation.

Project Details

The Grid-Occam project will study compiler construction on the Rotor platform. It will focus on developing compiler tools and a runtime environment to support programming-in-the-large for heterogeneous, adaptive systems. Using Rotor as foundation for the Grid-Occam project will allow for portability. Our research will impact the Software Engineering curriculum at HPI. Work on the project will be accompanied by a lecture for Master students and a series of lab experiments.

The Grid-Occam system will consist of the following components:

1. A Grid-Occam compiler, which transforms Occam programs into common intermediate language.
2. A choice of Grid-Occam runtime systems, for different infrastructure constellations, namely:
    a. A single-process implementation of Occam, where Occam processes are implemented as Rotor threads
    b. A distributed Occam implementation, where Occam processes are mapped to different Rotor application domains, using Rotor IPC mechanisms for communication.
    c. A Grid-Occam implementation, where Occam processes are executed in a computing grid using grid services (OGSA) for communication.
3. A configuration and deployment tool, supporting distribution of an occam application in an infrastructure, depending on the selected runtime system.
4. The project will develop a number of proof-of-concept Occam applications.

A number of design decisions must be made to complete this project, for which potential alternatives will be studied, and a selection be made. Such questions include

- What technology should the compiler use? Possible alternatives are:
    - Pure (hand-written) C++, like the Rotor C# compiler.
    - Generated C/C++ using a compiler generator such as YACC, Cocktail.
    - Pure (hand-written) C#.
    - Generated C# using a compiler generator (such as nYACC, or Coco/R for C#).

- What kind of communication/synchronization mechanisms should be used for each runtime target? In particular:
    - What Rotor-API should be used for the distributed implementation? (e.g. plain sockets, remoting, web services).
    - What API should be used for the grid computing implementation? (e.g. creation of an OGSI OCCAM channel service, usage of an OGSI tuple space service, using native binary channels).
- What kind of thread granularity should be used? As Occam supports dynamic parallelism, each PAR statement could be implemented either fully parallel, or sequentially. Several strategies of allocating Occam processes dynamically to threads need to be studied.

Academic Relevance

This project will produce the first implementation of Occam in common intermediate language, and also the first implementation to use a Grid computing infrastructure for execution. The strategies of existing Occam implementations need to be studied to find out whether they apply to Rotor and grid computing.

In the past, we have worked on various compilers, including a compiler translating the ITU-T Specification and Description Language (SDL) into Microsoft AsmL.NET; this compiler is automatically derived from the ITU-T language specification.

In addition, this project will make reuse of existing compiler technology of rotor as much as feasible. So far, no active community of compiler authors for .NET has been established, so we need to contact interested individuals. In order to support the Rotor community, we will be hosting the Grid-Occam web-site, which will be run in conjunction with sscli.net.

Experience

Today's commercial off-the-shelf computer systems (COTS) typically are being optimized for high performance which often leads to unsatisfactory user experience for internet-wide distributed systems. The research activity of our group is concerned with paradigms, design patterns and implementation techniques for enhancing middleware technology for predictable computing. Besides studying programming models and approaches for wide-area distributed computing (Grid), we currently investigate how far middleware technology can be pushed into the domain of embedded computing, thus linking embedded control systems with standard middleware.

A number of our research projects are centered around the configuration problem for component software, specifically the "online replacement of software components" and the use of commercial off-the-shelf operating systems in control systems ("Windows 2000/CE in real-time robotics and process control"). Projects are currently sponsored by Microsoft Research Cambridge and Deutsche Post ITSolutions. We are also participating in the DISCOURSE project of 4 universities in Berlin and Potsdam (Freie Universität

Berlin, Technische Universität Berlin, Humboldt Universität Berlin, Hasso-Plattner-Institut Potsdam). Our group is currently running a Grid computing testbed and Condor cluster consisting of approximately 50 nodes.

III. Project Plan

Deliverables/Milestones

During lifetime of the project (2 years) we expect to initiate and supervise roughly 4 Master/Diploma students during their mandatory project works and Master theses. The Rotor-based Grid-Occam Compiler will be used to teach concurrent and parallel programing in context of the Software Engineering curriculum at our institute.

We plan to publish results of our research in the proceedings of an international conference. A summary of the project's state including current findings will be published in a technical report (either via HPI/Potsdam University or MSR).

Intellectual Property

The project will develop a Rotor-based compiler and runtime-environment for Occam, which will be put in the public domain. Additionally, we will make the project's outcome available for demonstration in a testbed scenario at Hasso-Plattner-Institute at University Potsdam.

References

Wolfgang Schult and Peter Tröger,
Loom.NET - an Aspect Weaving Tool,
in Workshop on Aspect-Oriented Programming, ECOOP'03, Darmstadt 2003.

Richard Zurawski, Ed., Martin von Löwis and Peter Tröger,
Chapter on ".NET Technology" in Handbook on Industrial Information Technology
CRC Press LLC, 2003.

Richard Zurawski, Ed., Andreas Polze
Chapter on "Middleware" in Handbook on Industrial Information Technology
CRC Press LLC, 2003.

Andreas Prinz and Martin von Löwis,
Generating a Compiler for the SDL Formal Semantics Definition
Springer LNCS 2708, Axel-Springer Verlag, July 2003.

Andreas Rasche and Andreas Polze
Configuration and Dynamic Reconfiguration of Component-based Applications with Microsoft .NET
in Proceedings of International Symposium on Object-oriented Real-time distributed Computing (ISORC) 2003

Peter Tröger and Andreas Polze
Object and Process Migration in .NET
in Proceedings of Workshop on Object-oriented Dependable Real-time

Systems (WORDS 2003), Guadalajara, Mexico, January 2003, IEEE Computer Society Press, 2003.

Wolfgang Schult and Andreas Polze
Speed vs. Memory Usage - An Approach to Deal with Contrary Aspects
in Proceedings of the 2nd International Conference on Aspect-Oriented Software Development (AOSD2003)

Andreas Rasche and Andreas Polze
Configurable Services for Mobile Users
in Proceedings of Workshop on Object-oriented Dependable Real-time Systems (WORDS 2002), San Diego, USA, January 2002, IEEE Computer Society Press, 2002.

Occam resources: Simple Concurrent Systems Design and Development
http://wotug.ukc.ac.uk/occam/

Ian Foster, Carl Kesselman
The Grid: Blueprint for a New Computing Infrastructure
Morgan Kaufmann, 1st edition, 1998.