A process and memory model for a parallel distributed-memory machine

Petro Istavrinos Lothar Borrmann

Siemens AG, Corporate R&D, ZFE IS SYS 33
Otto-Hahn-Ring 6, D-8000 München 83, West Germany
Phone: + + 49 89 636 42350, E-Mail: borrmann@ztivax.siemens.com

Abstract

Currently there is no standard operating system suited for large parallel machines with distributed memory. Today's parallel programs have to be tailored to the peculiarities of a parallel machine like sequential programs in the bad old days of assembler programming.

Suitable abstractions for large scale parallelism have to be defined, which can serve as a basis for explicitly parallel programming as well as for very high level (declarative) parallel systems.

We defined a set of abstractions that will be implemented by the kernel of a large parallel machine. This includes a novel process model and a virtually shared store with customized coherency. The concepts will be used as a basis for parallel Lisp, parallel Prolog and a large parallel database system.

Introduction

The European Declarative System (EDS) is a project concerned with the development of a scalable multiprocessor technology and with the implementation of a prototype to demonstrate the system. Main application areas are parallel relational database servers, parallel Lisp and parallel Prolog.

The underlying hardware for EDS is a message-passing machine supporting up to 256 processing elements (PE) with large private memories. The PEs are connected by a high speed delta network.

An important component in the system is the operating system kernel providing an interface to the parallel machine and satisfying the requirements of the different execution models ranging from compatibility and flexibility of the interface to multiuser and multitasking facilities.
As complexity increases with advanced hardware capabilities and user requirements, the need for abstractions to hide this complexity also increases. The evolution of operating systems is no exception. The current state of the art is operating systems (such as Mach) for (shared memory) multiprocessor machines. According to the terminology of Young et al. [You87] the EDS hardware is a NORMA (No Remote Memory Access) machine in which processors have no way to access common physical memory. For this type of machine it is not possible to simply rely on any existing operating system without substantial modifications.

We decided to take an evolutionary approach for the development of the kernel and will stay in the tradition of operating systems evolving from Unix. We will use an existing operating system (Chorus [Roz88]) for distributed systems as a starting point.

The kernel interface of the EDS machine called Process Control Language (PCL) incorporates concepts that are present in those "state of the art" operating systems. However it also presents some concepts that are quite progressive and are not found in available systems. Among those is the organization and provision of mechanisms to support the abstraction of a virtually shared memory.

There has already been some research activity to provide the shared memory programming model also for distributed and parallel systems [Abr85, Bis88, Hsu89, Li86, Li88, Ram88, War88]. There is also a paper describing the design and implementation of a distributed shared memory server for Mach [For88]. The basic scheme is to introduce a cache-like copying protocol for memory objects (e.g. pages), allowing several processors to read remote addresses without any communication overhead on subsequent accesses. This may lead to a situation where multiple instances of one memory object exist, raising a coherency problem. The approach taken here allows to define customized protocols as required to deal with multiple instances of one memory object. The basic copying scheme is directly supported by the kernel.

The purpose of this paper is to give an overview of the PCL process model and then to focus on the capabilities which allow to emulate shared memory behavior.

**Process Model**

Commercially available today are Unix-derived operating system for shared memory multiprocessor machines with a limited number of processors (up to 32). Being limited to the abstractions of Unix (heavy weight user processes), they do not allow users to exploit all of the parallel capabilities of the underlying hardware.

The main drawbacks of Unix for parallel systems are the inadequate process model and inefficient inter process communication concepts. More suitable concepts