Reflective memory

Project Overview

Michael Pavlovsky and Gal Koren

April 9, 2003

Contents

0.1 Final goals

- Distributed, coherent, shared memory replicated over the local Ethernet network
- Programmer's API should resemble Linux shared memory semantics.
- Conformance to ACID rules (transactional behavior).

0.2 Milestones

- Overview (this document)
- Specification and background research
- Design and specifications refinement
- Implementation
- Testing and Integration
- Distribution

See more on the time schedule page.

0.3 Project background and motivation

0.3.1 A Few words about Shared Memory

You can safely skip this section if you have basic idea about what is all about. Please pay attention to ©note,below. [1]

What is Shared Memory? Shared memory (SHM) is another method of interprocess communication (IPC) whereby two or more processes share a single chunk of memory to communicate. The shared memory system can also be used to set permissions on memory, allowing for things like malloc debuggers to be written.

Types of Shared memory available Basically there are two different types of shared memory available for most flavors of UNIX. As you may have guessed, each of the two original ancestors of modern UNIX have their own implementation, although almost all modern UNIX flavors implement both. The names of the respective implementations are System V IPC, and BSD mmap.

General Principles Basically using shared memory under both systems will involve the following:

Uniquely naming the segment Each implementation has a method establishing a system wide unique name for each segment so that other applications may access it. Both also have methods of procuring private (or "anonymous") segments.

Specifying access permissions Both implementations allow you to specify access using the traditional read/write/execute scheme.

Race conditions Dealing with race conditions varies under each system. What they do have in common is that race conditions do occurs in each :)

0.3.2 DSM Distributed Shared Memory

Distributed shared memory (DSM) systems represent a successful hybrid of two parallel computer classes: shared memory multiprocessors and distributed computer systems. They provide the shared memory abstraction in systems with physically distributed memories, and consequently combine the advantages of both approaches. Because of that, the concept of distributed shared memory is recognized as one of the most attractive approaches for building large-scale, high-performance multiprocessor systems. The increasing importance of this subject imposes the need for its thorough understanding.

Distributed Shared Memory enables programs to access data in traditional virtual memory . It is primarily a tool for parallel application or a group of applications in which individual shared data items can be accessed directly.

[1] I hope I took this paragraph from one of the Linux How-Tos. If you think differently please correct me.

In systems that support Distributed Shared Memory, data moves between secondary memory and main memory as well as between main memories of different nodes. Each node can own data stored in the shared address space, and the ownership can change when data moves from one node to another. When a process accesses data in the shared address space, a mapping manager maps the shared memory address to the physical memory. The mapping manager is a layer of software implemented either in the operating system kernel or as a runtime library routine.

The Distributed Shared Memory spares the programmer the concerns of message passing when writing applications that might otherwise have to use it. Shared memory provides the fastest possible communication, hence the greatest opportunity for concurrent execution.

0.4 Used technologies, license schema, software and hardware requirements

0.4.1 Software Requirements

Important

- First of all, until implicitly mentioned, non of this is not supposed to be pre-installed.
- Second,since this document is only an overview I allow myself to quote a big parts from used software's documentation.

Networking and threads From my experience the best solution is ACE. I am quoting:

> The ADAPTIVE Communication Environment (ACE) is a freely available, open-source object-oriented (OO) framework that implements many core patterns for concurrent communication software. ACE provides a rich set of reusable C++ wrapper facades and framework components that perform common communication software tasks across a range of OS platforms. The communication software tasks provided by ACE include event demultiplexing and event handler dispatching, signal handling, service initialization, interprocess communication, shared memory management, message routing, dynamic (re)configuration of distributed services, concurrent execution and synchronization.
>
> ACE is targeted for developers of high-performance and real-time communication services and applications. It simplifies the development of OO network applications and services that utilize interprocess communication, event demultiplexing, explicit dynamic linking, and concurrency. In addition, ACE automates system configuration and reconfiguration by dynamically linking services into applications at run-time and executing these services in one or more processes or threads.

ACE continues to improve and its future is bright. ACE is supported commercially via the Riverace corporation using an open-source business model. In addition, many members of the ACE development team are currently working on building The ACE ORB (TAO).

From ACE FAQ:

Won't I have to give away my source code if I use ACE? 1. No. ACE is not governed by the GNU Public License (GPL). It has its own usage and copying agreement. You can find it in the ACE source kit, in the top level directory, in a file named COPYING. The file in the kit is the one which governs your use of ACE, but you can also read a copy of it here. In short, you can use ACE for whatever you want - just don't try to claim ownership or copyright of it.

Graphs and basis for Distributed Algorithms I am going to make heavy use of GTL, its license is more problematic ,but since I am not heaving plans to sell this software it must be ok to use it. `http://infosun.fmi.uni-passau.de/GTL/`

GTL, the Graph Template Library GTL can be seen as an extension of the Standard Template Library STL to graphs and fundamental graph algorithms.

Graphs are a common model for discrete relational structures, such as diagrams, nets and networks, hierarchies, plans, charts, or maps. And there are many important algorithms for graphs, such us exhaustive search, shortest path, minimum spanning tree, matching or network flow, with a wide range of applications in areas such as Computer Science, Discrete Mathematics, Chemistry, Molecular Biology, Operations Research, Engineering, Computer Aided Design, Project Management, Production Planning, Surveillance, and Maintenance Control, Cartography, and other application areas. The wide use of graphs is also due to the fact that they have a natural graphical representation and Graphlet is a tool for this.

GTL is a platform independent and extendible C++ library. It is a useful aid to any software designer. GTL contains the classes needed to work with graphs, nodes and edges and some fundamental algorithms as building blocks for more complex graph algorithms. Further algorithms are under development.

Customized algorithms can be developed by experienced programmers. A major step shall be support for models of hierarchically structured graph and the efficient manipulation of really large graphs.

GTL has been designed and programmed following the guide lines of STL.

License The Graph Template Library (GTL) is copyrighted by the University of Passau. The following terms apply to all files in the GTL distribution:

The authors hereby grant permission to use, copy and modify this software and its documentation for any purpose, except the ones mentioned in this copyright notice, provided that existing copyright

> notices are retained in all copies. No written agreement, license, or royally fee is required for any of the authorized uses.
>
> Commercial use of GTL without prior written permission is prohibited. GTL must not be distributed without prior written permission. The terms of this copyright notice also apply to any modification to this software.

Data storage and memory pool I order to stand the goal of fast,reliable application and to improve speed of the development cycle I am going to use C++ with garbage collector. I addition to this I expect *bursty* load behavior therefore I want to manage amount of memory required bu running process. All above leads to smart embedded data storage like Berkeley-DB. Fortunately, mysql.com have released small embedded library just what I needed. Please take a look:

> The MySQL database server is the world's most popular open source database. Its architecture makes it extremely fast and easy to customize. Extensive reuse of code within the software and a minimalistic approach to producing functionally-rich features has resulted in a database management system unmatched in speed, compactness, stability and ease of deployment. The unique separation of the core server from the storage engine makes it possible to run with strict transaction control or with ultra-fast transactionless disk access, whichever is most appropriate for the situation.
>
> **License.** The MySQL database server is available without a license fee under the GNU General Public License (GPL). Commercial non GPL licenses are available for users who prefer not to be restricted by the terms of the GPL.
>
> **Embedded database library.** Using the embedded database library (libmysqld), you can include the full power of the MySQL database server into applications and electronics devices, without your end-user having any awareness of the underlying database. The embedded MySQL database is ideal for use behind the scenes in Internet appliances, public kiosks, turn-key hardware/software combination units, high performance Internet servers, self-contained databases distributed on CD-ROM, and more possibilities just waiting for you to invent them.

0.4.2 License schema summary

For moment being, I can happily live with GPL license.

0.4.3 Hardware Schema

This project is developed on the next hardware schema: **Three Pentium III class workstations with at least 128 Mb of memory. 10/100 Ethernet switch.**