Technical Overview of the CHORUS System

The Integration of Real-time, Distributed Processing and UNIX

The CHORUS technology integrates inter-process communication into the heart of it's real-time operating system. It is this integration that gives CHORUS it's essential properties: real-time; transparent distribution of, and access to, programs and files over a network or bus; modularity; and the efficient integration of UNIX and other open system standards into full real-time distributed operating systems.

The key characteristics of the CHORUS system are:

— *Real-time*: the CHORUS system is built on a real-time nucleus that has all the functional characteristics and performance of classic real-time executives.

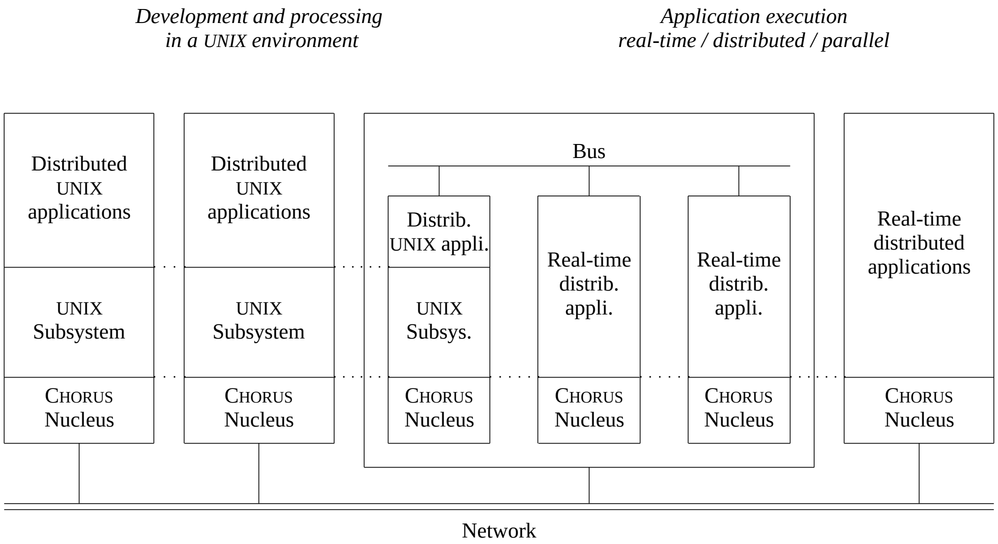
— *Controlled transparent distribution* of processing and of data: the management of distribution can be entirely taken care of by CHORUS yet controlled by network administrators. In particular, CHORUS permits dynamically reconfiguring the system and applications.

— *Modularity*: the CHORUS system is composed of a set of communicating modules that can be assembled and reconfigured dynamically depending on the capabilities of the hardware and the needs of applications.

— *Compatibility* with UNIX: application programs, utilities, and files used under UNIX can be supported without modification (binary compatibility) by the CHORUS system.

— *Openness*: CHORUS provides all the functionality of a standard UNIX system (System V - X/OPEN), but can also host other operating system interfaces. In addition, CHORUS supports the communications standards OSI and TCP/IP.

The immediate domains of application for CHORUS are those for which the limits of traditional operating systems are preventing the development of fully satisfactory solutions. This includes especially the development of advanced real-time systems, of industrial and technical systems (process-control, command stations, computer-aided automation, test and measurement, telecommunications, and military systems) and specialized systems (dedicated servers, multi-processor and parallel machines).



The CHORUS Nucleus and subsystems

Structure of the CHORUS system

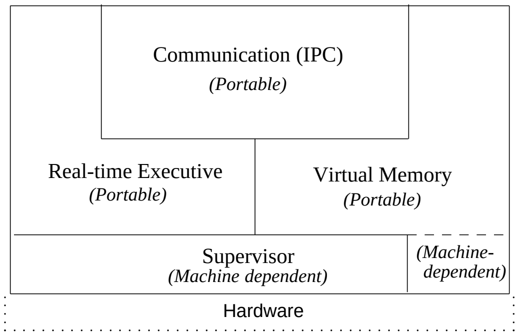
CHORUS is composed of a small real-time distributed *Nucleus* (on the order of 100K bytes) and one or several *subsystems* offering application programs standard operating system interfaces. The UNIX subsystem is the first of these subsystems.

The Nucleus

The CHORUS Nucleus is built around a *real-time executive* that supplies the minimum set of functions that are needed to support of the smallest system configurations. The real-time executive provides multi-tasking (CHORUS *threads*), scheduling threads by their priorities, managing the preemption of running threads by interrupts and primitives for the synchronization of and low-level local communication between threads.

The CHORUS real-time executive is normally augmented with:

- A low-level hardware supervisor permitting the dynamic association of interrupt handling code to interrupts.
- An Inter-process communication (IPC) system for communicating by exchanging *messages* between *ports* attached to groups of threads in a given address space (CHORUS *actors*). This IPC system offers a uniform vision of communication that is transparent to the distribution (location) of the communicating threads.
- A distributed *virtual memory* manager for those applications having such a need and running on processors equipped with the necessary hardware.

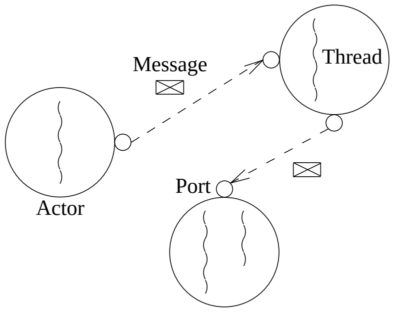


Structure of the CHORUS Nucleus

Actors and Threads

The CHORUS Nucleus permits the execution of *threads* (an execution context) that can share the resources of the same *actor* (address space). The threads communicate between themselves independently of their location by using the CHORUS IPC. Privileged "system" threads may be installed directly into the Nucleus address space.

Threads and actors are used in exactly the same way to implement system functions as to support user processes.



Actors, threads, ports and messages

Subsystems

The basic services of the CHORUS Nucleus can be enriched by adding *subsystems* that supply to application programs a complete operating system interface. A subsystem is built from a set of CHORUS actors playing the role of servers to provide their services to application programs.

The first subsystem available with CHORUS is a UNIX subsystem offering a standard UNIX (System V - X/OPEN) environment, extended to be a distributed system while respecting real-time constraints for local operations.

The CHORUS Nucleus

The CHORUS Nucleus offers the functionality and performance of a classic real-time executive.

Thread management

The CHORUS Nucleus permits you to create and delete threads, to suspend and resume their execution, etc....

The scheduling of threads is preemptive. Each thread receives a priority level at it's creation; when several threads are ready to be executed, the Nucleus choses the thread with the highest priority; for threads of equal priority, the threads are scheduled in FIFO (first-in, first-out) order. The Nucleus reschedules each time an interrupt or exception is produced and each time a thread makes a system call to the Nucleus that changes the current thread (e.g., suspend, ...).

Structuring of threads

The CHORUS Nucleus groups threads into *actors* that provide to each of their threads a private execution context (address space) and communication mechanisms protected from the intervention of threads exterior to the actor.

Communication and synchronization

CHORUS offers two modes of communication and synchronization:

— The threads of a given actor share that actor's address space. They can communicate directly among themselves by using shared memory. CHORUS includes synchronization mechanisms that can be used to ensure the proper coordination of threads so synchronized:
  - mutual exclusion semaphores,
  - execution control (suspend / resume) of a thread.

— CHORUS also provides a message-based communication mechanism that can be used between any thread – in the same actor or in other actors, regardless of which machine they may be on in a network. The principle characteristics of this communication service are:
  - the communication interface is independent of the localization of the threads and thus permits configuration changes without having to modify the code of the threads,
  - messages are exchanged between *ports* or *groups of ports*, with addressing modes specially adapted to distributed applications and without reducing the performance of local communication.

  The exchange of messages is equally a method of synchronizing threads, whether the threads are located in different actors or in the same actor.

Memory management

CHORUS offers mechanisms for allocating and deallocating physical memory.

When the hardware permits and the applications need it, CHORUS can also manage *virtual memory*. Specified pages can be locked into physical memory in order to guarantee the response time of threads operating under strong real-time constraints.

Interrupt handling

CHORUS allows connecting user-written routines to machine interrupts; these routines can be installed and replaced dynamically and do not need to be linked into the Nucleus. This ability is useful for debugging and maintenance as well as for adapting to hardware reconfigurations.

The interrupt routines can trigger (signal or start) threads which are then scheduled by the Nucleus according to their relative priorities.

The UNIX subsystem

CHORUS supplies a standard UNIX environment, integrated with the CHORUS real-time environment described above, and conforming the X/OPEN specifications. The standard UNIX services have been extended to the distributed environment (distribution of programs as well as files) and to real-time in a manner that is completely transparent to UNIX application programs.

Compatibility with UNIX

CHORUS is compatible with UNIX at three levels:

— *system call compatibility:* CHORIX is compatible with UNIX system calls at two levels:
  - compatibility at the source code level for all programs written for UNIX and following the X/OPEN base level standard.
  - compatibility at the executable binary code level for certain reference UNIX machines serving as porting bases;

— *file compatibility:* CHORUS offers total compatibility with UNIX files. Files created under UNIX can be used just as they are in the CHORUS environment.

— *device driver compatibility:* CHORUS offers functional compatibility with UNIX device drivers, reducing considerably the cost of porting CHORUS onto machines already having a UNIX system.

Signal management

CHORUS offers the UNIX set of functions for managing signals, but extends them to the management of real-time processes and to distributed processes. This extension is transparent and permits the remote creation of processes on any machine, while respecting the inheritance rules of UNIX (environment, open files, etc.), and permits "manipulating" those processes at a distance.

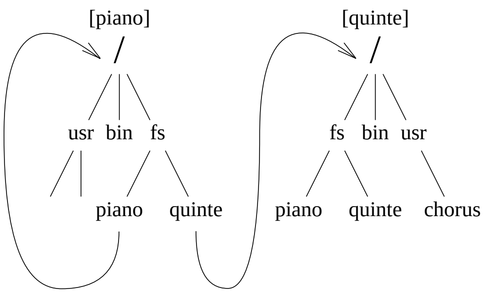
CHORUS provides the traditional UNIX communication mechanisms between processes and extends them to a distributed environment in a manner that is transparent to applications. Furthermore, CHORUS provides a mechanism for message-based communication between processes (IPC) that is particularly well adapted to distribution.

File management

CHORUS offers the UNIX set of file management functions and extends them, transparently, to supply a true *distributed file system*:

— *The automatic interconnection of local file trees* permits accessing any file located on any machine on the network.

— *Symbolic links* permit masking the distribution of files and their placement, to "personalize" naming and to be able to reconfigure systems in a manner transparent to applications and users.



Interconnection of file trees

Utilities

CHORUS supports the UNIX utilities defined by the X/OPEN Portability Guide – for program development, the shell, for file manipulation, etc. – and extends them to distribution. So, for example, the file manipulation utilities offered by CHORUS permit manipulating distributed files as if they were local UNIX files, without being concerned with their location.

Development environment

The integration of the standard UNIX environment and its extension allow CHORUS to offer a software development environment that is particularly rich, including utilities such as:

- editors,
- configuration management systems,
- symbolic debuggers extended to debug distributed processes,
- *CHORUS System Simulator* on a native UNIX,
- etc.

CHORUS: an open system

CHORUS is a true open system:

- CHORUS supports the international open standards, not just in terms of the system interface (UNIX standard), but also in terms of communications* (OSI/TOP, TCP/IP).

- CHORUS supports the traditional UNIX protocols that permit exchanges between UNIX machines.

The CHORUS technology, because of its modularity and clear organization, is open to extension and evolution: to support future open system standards and new machine architectures, to adapt to particular user needs, to integrate new functionality, etc.

* The CHORUS support for communication standards is implemented using the *OSIAM* protocol management software from Marben.

Reorder part number: CS/TR-89-10.2.0