Distributed Systems

Overview

Impact of moving from a single machine to a networked set of machines has been tremendous. Look at Fig 9-1 for comparison of centralized vs. distributed. Also compare to isolated computers (9-2). Disadvantages (9-3).

Distributed System— "Autonomous machines linked by a network with software designed to produce an integrated computing facility."—CDK.

"A collection of independent computers that appears to its users as a single coherent system."—Tanenbaum

"A collection of loosely coupled processors interconnected by a communication network."—SGG

Hardware

Different types of hardware in which to build systems:

- multiprocessor—a single address space among the processors.
- multicomputer—each machine has its own private memory.

Can develop an operating system for either type of environment.

Two types of bus-based multiprocessor organization:

1. *Uniform Memory Access (UMA)* model. Show picture.

   Caching is vital to get reasonable performance. For example, caches on a shared memory multiprocessor.

   Want to maintain *cache coherency. write-through cache*—any changes to the cache are written through to memory. Combine with other processors on the bus watching the bus (*snooping or snoopy cache*).

   Also can have *write-back cache*—only write the changed contents back to memory if another request is made.

2. *Non-Uniform Memory Access (NUMA)* model. Show picture. Or a hierarchy where CPUs have their own memory (not the same as a cache). Access costs to memory is non-uniform (NUMA). Use memory locally or copy to our own machine.

A multicomputer has a non-uniform memory access model as well, but do not have a common address space. A *distributed shared memory system* at least layers a common address space.

Software

The hard part to make the system work. Approaches:

- *distributed operating systems*. Tightly coupled operating system for homogeneous multicomputers. Goal is to make distributed system appear as a single machine.
- *network operating system*. Loosely couple operating system for hetergeneous multicomputers on in LAN/WAN environment. Allows sharing of some services and resources. A distributed file system, such as Sun's Network File System (NFS). is a good example.

  All accomplished with *servers* on the remote machine. Processes waiting to handle requests. Systems are heterogeneous and autonomous (make own decisions).
- *middleware systems*. Additional layer on top of NOS (or traditional operating system implementing general-purpose services. Provides access transparency. CORBA, DCE, DCOM, Jini.
- Distributed applications such as WWW, gopher, netnews. Not an entire DOS, but needs underlying support.

See Tanenbaum Fig 1-24 for comparison.

Design Issues

Resource Sharing

Fundamental use of networked computers (why network them otherwise?). Files, information, work (computer supported cooperative working (CSCW)). Need policies and mechanisms for sharing the resources. Have clients and servers of information.

Can also be done with the object model.

Transparency

- location—unaware of resource location (NFS, not Web)
- access—identical operations used for local and remote resources (URL)
- migration—resources can move without changing names
- replication—system can have any number of copies
- concurrency—protection of shared resources in face of parallel users (printer sharing)
- failure—concealment of faults
- persistence—user may view all objects as persistent occur

Openness (Flexibility)

Can the system be extended?

Need to design it into the system and publish the interfaces. Unix was an early open system. Look at DCE, CORBA and Jini as standards for creating open systems.

Reliability

One goes down, have high probability that other machines are available. But distributed systems often have dependencies on one or a few machines.

Leslie Lamport on a distributed system "One on which I cannot get any work done because some machine I have never heard of has crashed."

Fault tolerance (ability to recover from faults)—hardware and software solutions.

Can we detect errors—what's the difference between a communication error and a computation that takes a long time to complete?

Issue of availability (how much is the system usable). Did a workstation or a server machine crash?

Performance

Different measurements. Actual applications versus low-level benchmarks.

Scalability

How big in terms of machines and distance? Affects resource management and location. A distributed file system vs. Internet as an example. Affects issues of caching and replication.

Should be able to extend any resource.

Naming

Need an identifier for an entity. Often two levels of names: those used by users and those used by the system.

Structured or flat?

Name service to resolve names to their underlying value.

Names have contexts

Communication

Paradigms:

- message passing
- remote procedure call
- group communication

Computation Speedup

Approaches:

- Processor pool (dedicated computation servers).
- Use of idle workstations (absorbtion of idle power).
- Shared memory multiprocessor.

Consistency

1. update (by multiple applications)
2. replication (replicated copies consistent)
3. cache (cached copies consistent—Web)
4. failure
5. clock
6. user interface