UNIX
Batch Systems

Roberto Innocente

2000/06/20

Batch systems - main streams

* Load sharing/Load balancing systems : their object is
  fair sharing among multiple tasks of a
  single resource and/or balanced distribution of tasks
  among multiple nodes (NQS and its derivatives ,LSF)

* Checkpointing/job migration systems: their purpose is to
  use personal workstations/PCs while they are
  idle, stopping and resuming or migrating jobs
  between machines as the need arises (Condor)

* Mixed : some systems try to follow both paradigms

Batch systems - Load sharing/load balancing systems

With these systems jobs are usually executed under the uid and gid of the submitter.

Sharing of files between submitting and executing machines
is usually required. This is done via some standard file sharing mechanism like:
NFS or AFS or DFS.

The standard shells can usually be used as job scripting languages.

NQS example :

```
# nqs example
echo `pwd`
ls
a.out <inp.file >out.file
```

Batch systems - Checkpointing/Job migration systems /1

With these systems usually the job is run under a single uid/gid (e.g. condor/condor) to avoid the security implications and management complications connected with sharing user and groups between machines across administrative boundaries.

System calls (open, read, ...) are intercepted on the executing machine (linking the object file with a special library) and forwarded to a shadow process on the submitting machine that performs them. In this way no file sharing is necessary.

Only a restricted kind of executable (idempotent I/O, no long time sockets...) linked with the checkpointing library can be migrated, therefore these systems have a special job language and they do not usually accept shell scripts.
Condor example :

```
#  condor example
Executable = a.out
Input = inp.file
Output = out.file
Requirements = Memory > 128
Queue
```

Batch systems - Checkpointing/Job migration systems /2

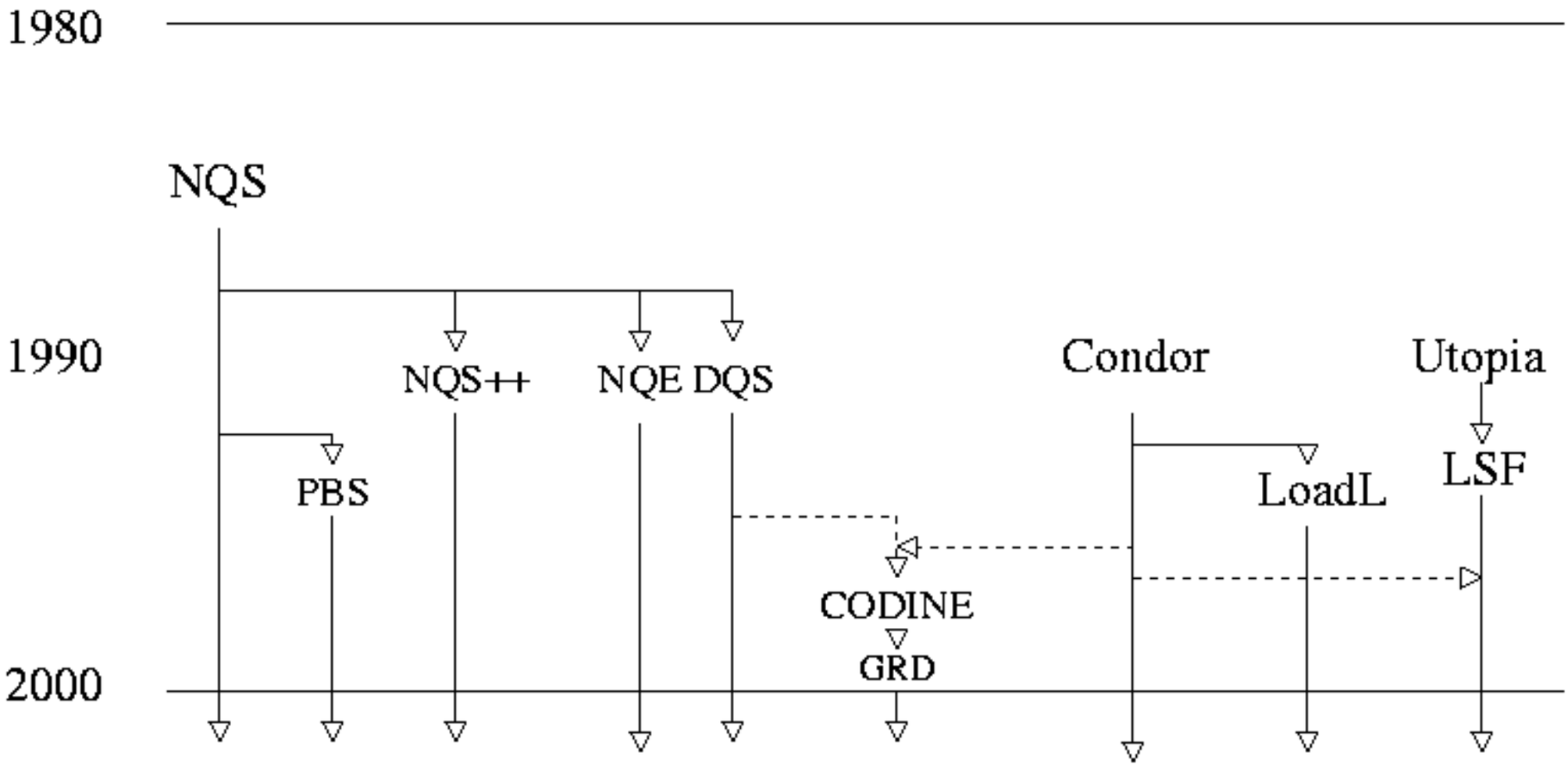
Requirements for checkpoint-ability (with Condor) :

* Idempotent I/O : files are opened read only or write only, not r/w.
  In this case re-execution of a read or write operation does'nt change the file state
* No long lasting socket ops : checkpoint creation will be delayed until socket operations will be completed
* No multiple process : fork()/exec() sys calls are not allowed
* No IPC : pipes, semaphores, shared memory not allowed
* No alarms, timers, sleeping: sleep(), alarm(), getitimer() are not allowed
* No memory mapped files.
* On Linux , Digital Unix and HP-UX: statically linked !!!

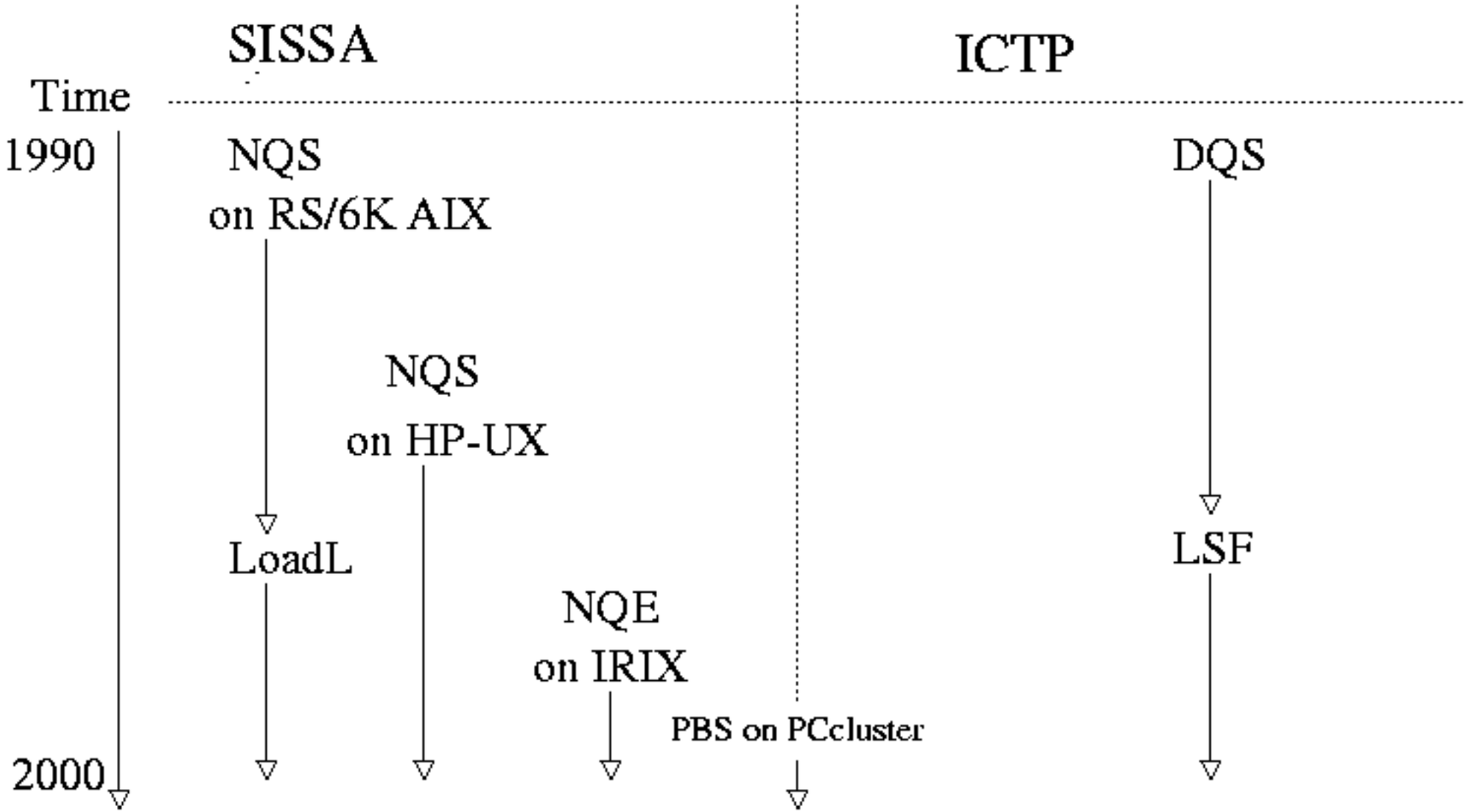
Batch systems - keywords

* heterogeneous environment (using $ARCH variable...)
* checkpointing (linking with special library)
* load balancing (static/dynamic)
* job migration (condor)
* parallel support (usually using a machine list)
* run-time/wall-time limits (with daemon support)

Batch systems - History and relationships



Batch systems - Adoption in SISSA/ICTP



Batch systems - Vendors/Software

Software proposed by system vendors :

IBM	LoadLeveler
HP	TaskBroker /LSF
Compaq/DEC	LSF
SGI/Cray	NQE
Sun	LSF

*loadL sold by IBM
*LSF sold by Platform Comp.
*NQE sold by SGI/Cray
*Task Broker sold by HP

On today unixes there is widespread support of :

SIGSTOP (SVR4/Bsd4.3+/Posix) stops a process (cannot be caught)
SIGTSTP (SVR4/Bsd/Posix) stops a process

SIGCONT (SVR4/Bsd4.3+/Posix) continue a process that was stopped

SIGXCPU (SVR4/Bsd4.3+) sent to a process when it exceeds its soft cpu limit

SIGXFSZ (SVR4/Bsd4.3+) sent to a process when it exceeds its soft file size limit

SIGUSR1, SIGUSR2 (SVR4/Bsd) user defined signals

Widespread support of resource limits using (Svr4/Bsd) :

struct rlimit {int rlim_cur; int rlim_max};

int getrlimit(int resource,struct rlimit* rlptr);
int setrlimit(int resource,const struct rlimit *rlptr);

where resource can be :

RLIMIT_CORE (Svr4/Bsd) maximum size in bytes of a core file
RLIMIT_CPU (Svr4/Bsd) max amount of cputime in seconds
RLIMIT_DATA max size in bytes of data seg
RLIMIT_FSIZE max size in bytes of a file
RLIMIT_MEMLOCK (Bsd) max size of locked memory
RLIMIT_NOFILE (Svr4) max # of open files
RLIMIT_NPROC (Bsd) max # of child process
RLIMIT_OFILE (Bsd) same as _NOFILE
RLIMIT_RSS (Bsd) max rss size
RLIMIT_STACK max stack size
RLIMIT_VMEM max mapped address space

Batch systems - Unix support : resource usage

The following call, returning info
on resources used, is also supported by the SVR4/bsd4.3+ :

```
int getrusage(int who,struct rusage *rusage);
```

where who is RUSAGE_SELF or RUSAGE_CHILDREN and

```
struct rusage {
        user time
        sys time
        max rss
        integral shared mem,data mem,stack
        page reclaims, faults
        block i/o operations
        messages sent/received
        signals received
        context switches voluntary/involuntary
}
```

Batch systems - Unix support : setjmp()/longjmp()

These functions allow a non-local goto (they are used for example by Condor in its checkpoint/restore mechanism):

```
#include <setjmp.h>

int setjmp(jmp_buf jmpbuf);
void longjmp(jmp_buf jmpbuf,int val);
```

On i386 architecture an array of six integers is stored, with the following contents :

BX, SI, DI, BP, SP(=stack pointer), PC (=program counter)

plus the signal mask.

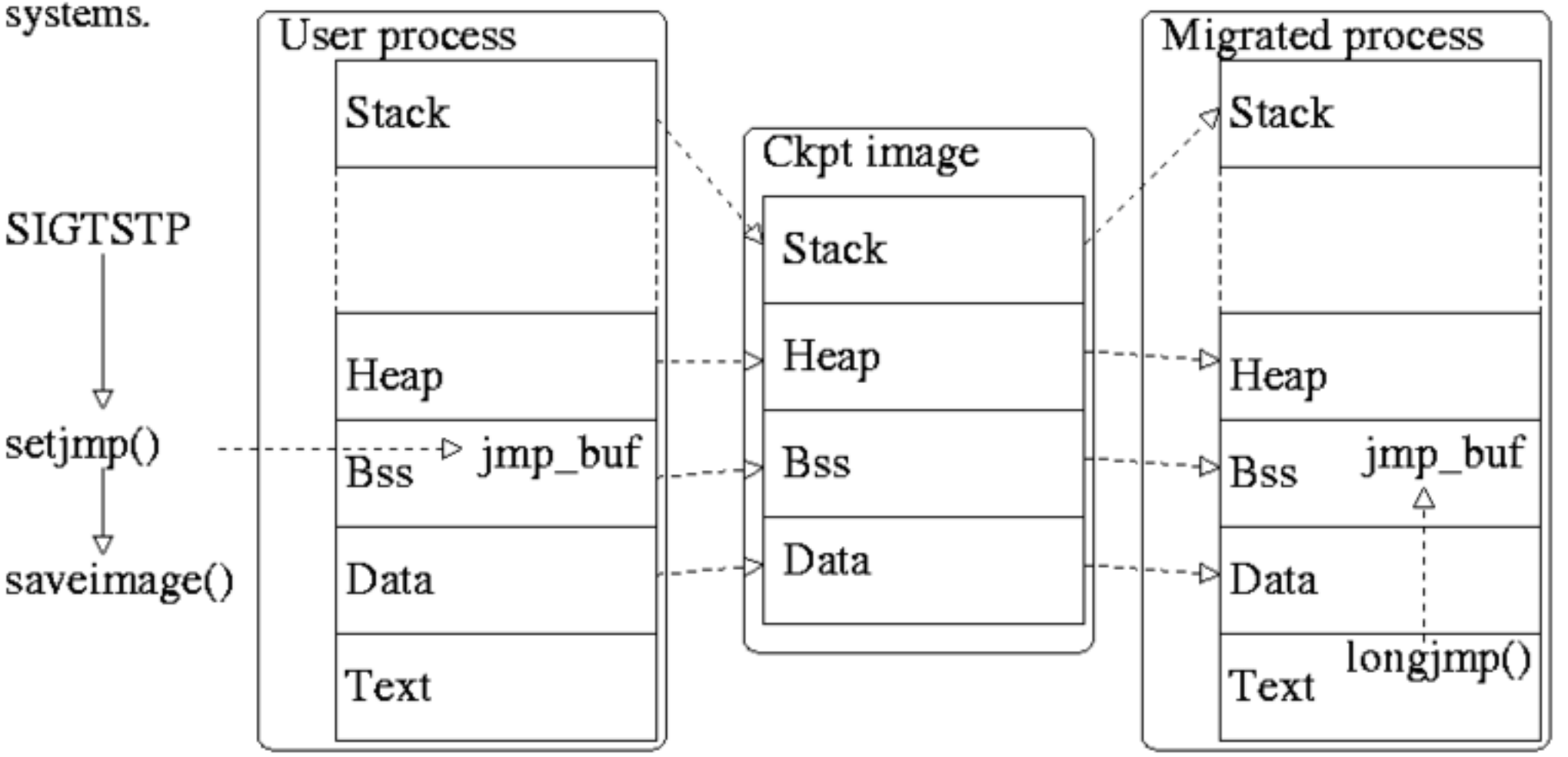
Example :

```
if (setjmp(jmpbuf) == 0) {
    /* first time through */
else {
    /* coming from longjmp()  */
}
```

N.B. : It is better to use the POSIX equivalents sigsetjmp/siglongjmp for which the behaviour respect to the signals and signal mask is clearly stated.

Batch systems - Checkpointing/Migration details /1

Checkpointing is the ability to take a snapshot of the full status of a running job in such a way that the job can be restarted at the same exact point.
If this can be done transparently and in an automatic way across different machines on a network then we speak of dynamic load balancing/job migration systems.



Batch systems - Checkpoint/Migration details /2

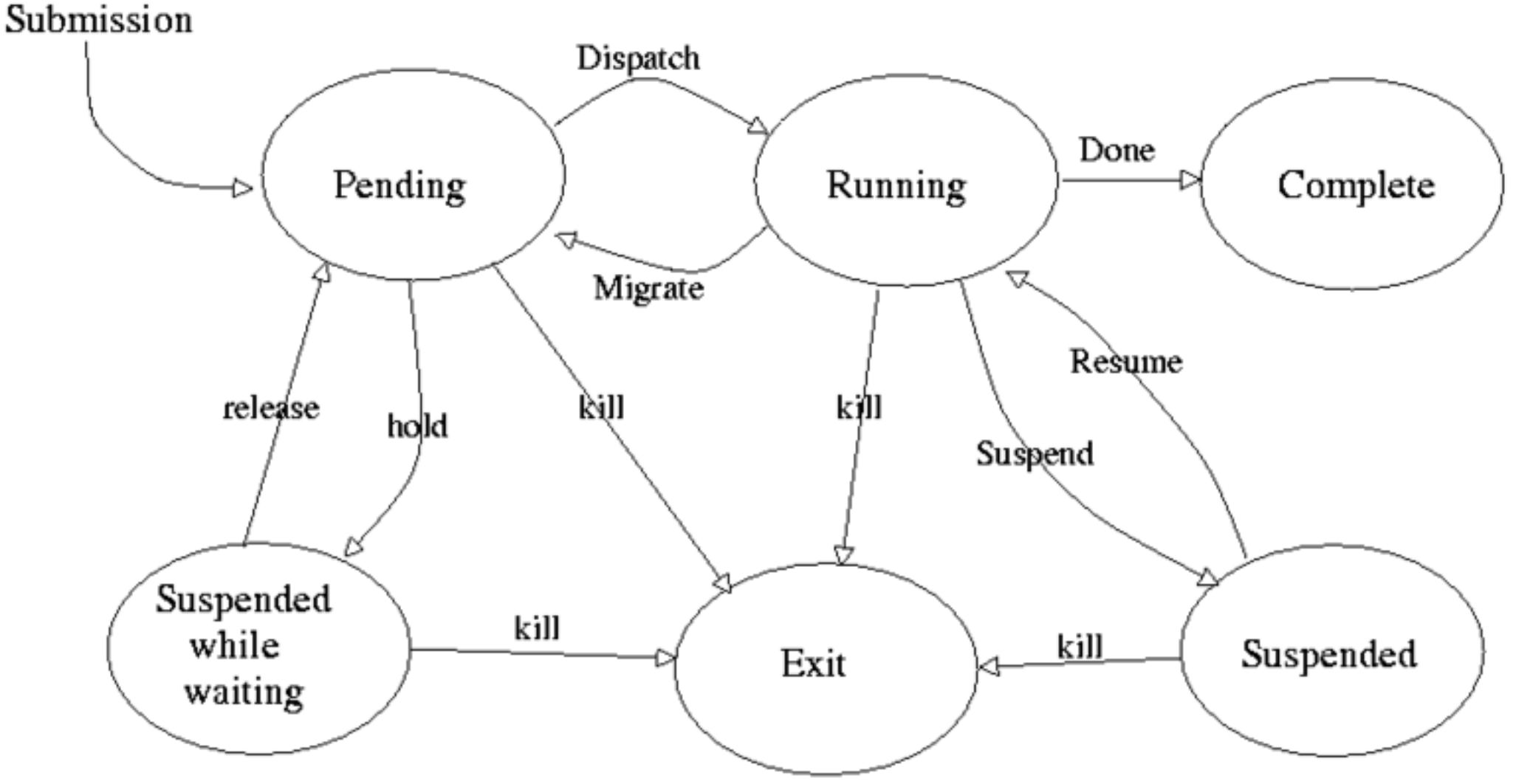
Detailed steps :

- When the program starts checkpoint() is installed as signal handler for the SIGTSTP signal
- When necessary a SIGTSTP signal is sent to the user-process requesting a vacation
- The process calls the checkpoint() routine and saves in its data area :
  - * a buffer describing the status of the machine obtained with setjmp(jmp_buf)
  - * a table of currently opened files, and their current pointer
- The process writes on a checkpoint image the data and stack segment and terminates itself with USR2
- The checkpoint image is migrated to another computer
- The executable is restarted with special arguments so that the restart() function of the Condor library is invoked
- Data and stack segments are copied from the checkpoint image, files are reopened and repositioned, signals and handlers are re-established
- longjmp(jmp_buf)

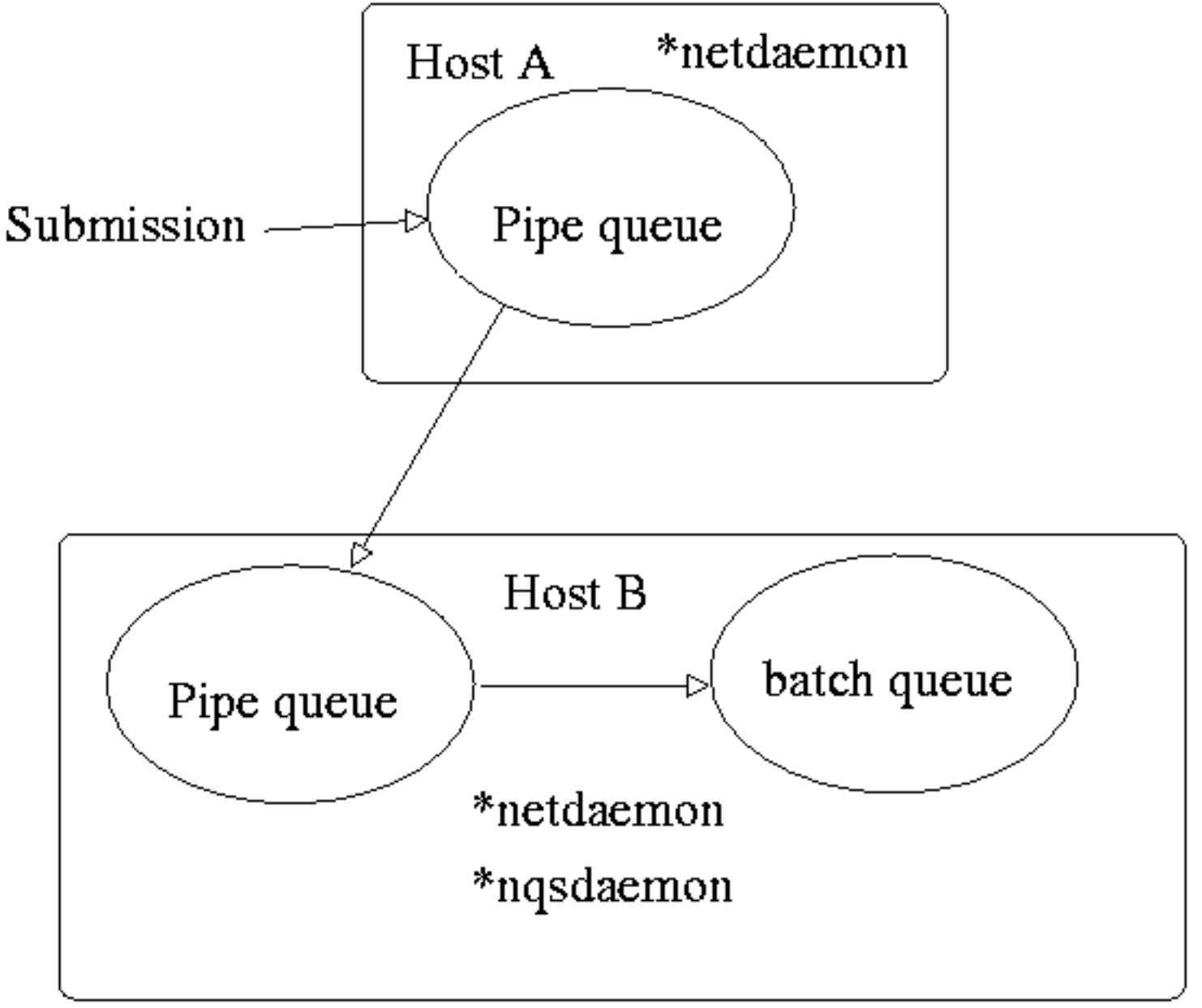
Batch systems - command comparison

NQS	LoadL	NQE	LSF	Condor
qsub	llsubmit	qsub	bsub	condor_submit
qstat	llq llstatus	qstat	bjobs lshosts	condor_q condor_status
	llstop llresume		bstop bresume	
qdel qkill	llcancel	qdel	bkill	condor_rm

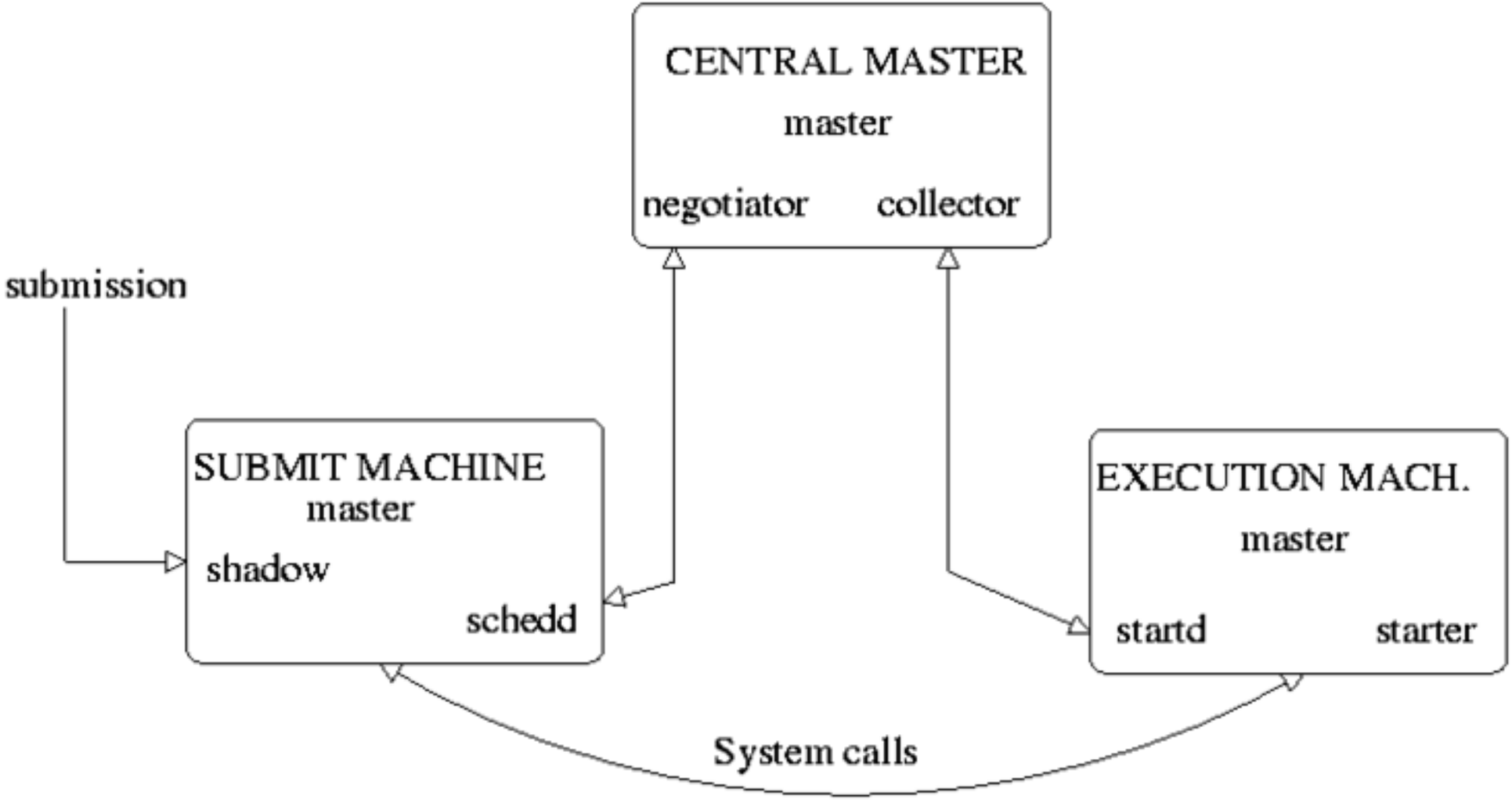
Batch systems - Typical state diagram



Batch systems - NQS (Sterling/Cosmic ...)

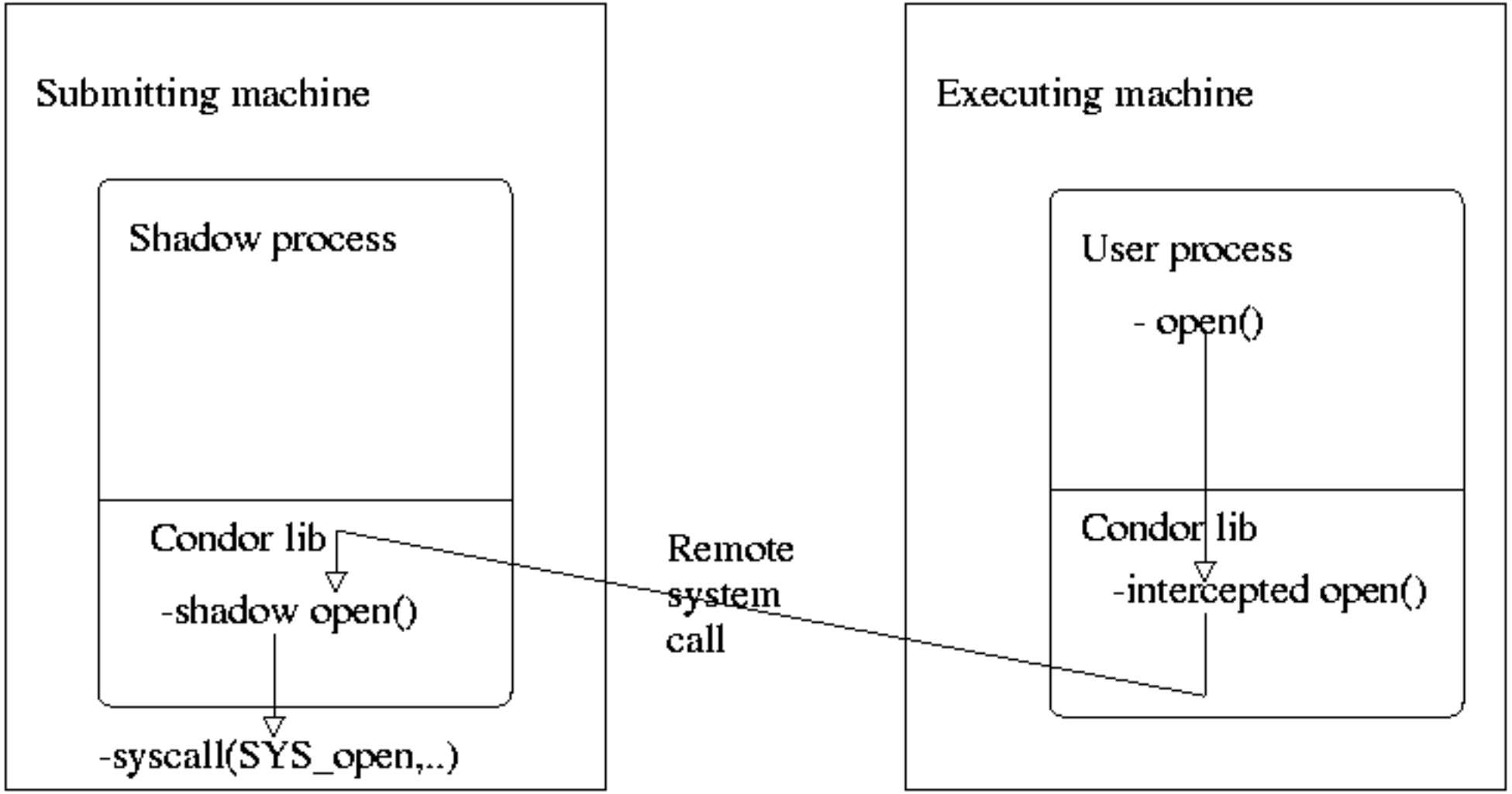


UPPERCASE = MACHINE DENOMINATION
lowercase = daemon

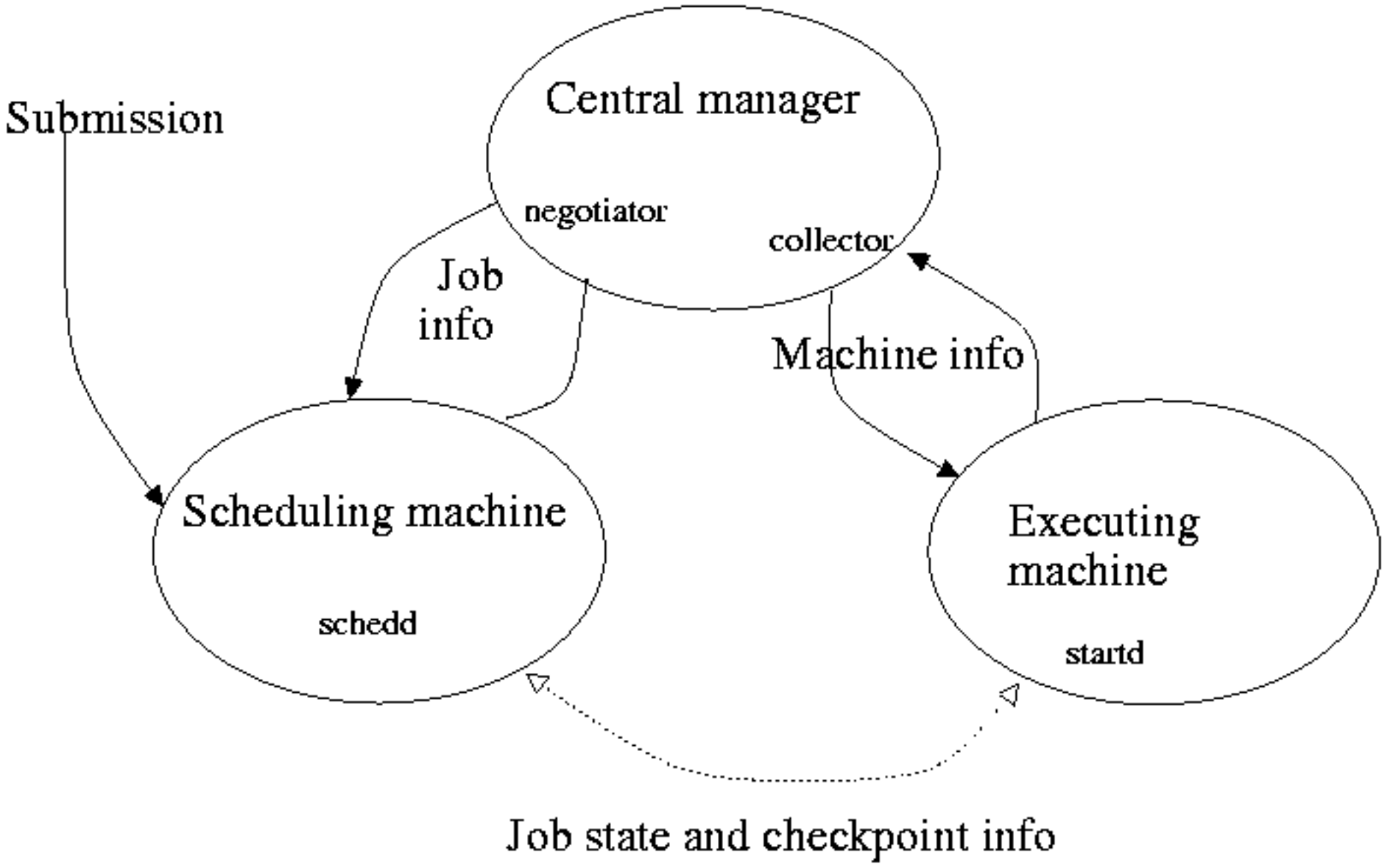


*Condor can manage std shell jobs in its "vanilla universe"

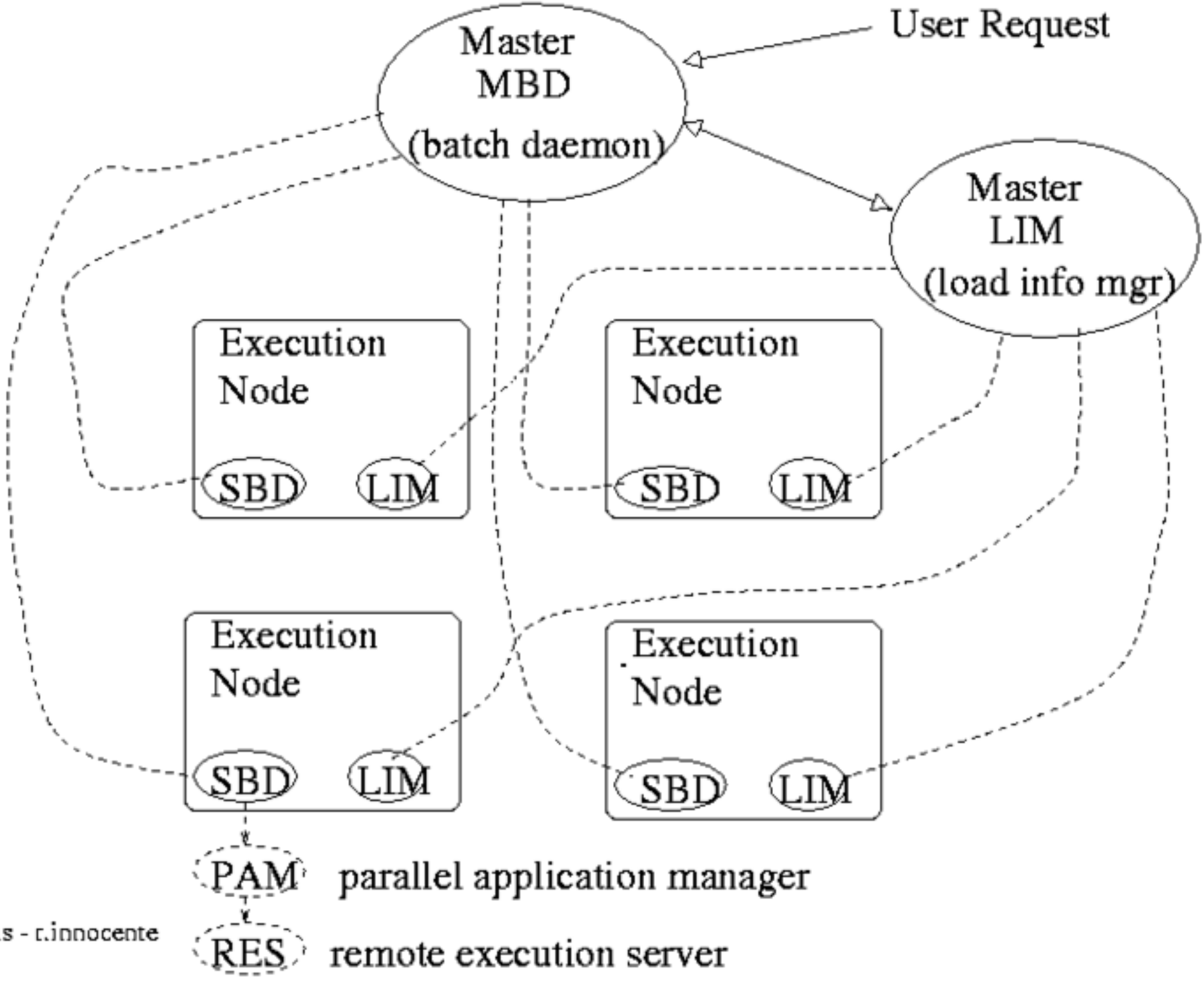
Interception/forwarding of system calls



Batch systems - LoadL (IBM)



Batch systems - LSF (Platform Computing)



Batch systems - Conclusions

- PBS is a stable and widely diffused system that supports parallel-mpi programs on clusters. We have used it some time last year and for about 6 months this year on a myrinet based PC cluster. I suggest it should be used for dedicated resources : servers, computing clusters

- Condor should be used to take advantage of the extremely high percentage of idle time of personal WKS and PCs

- CODINE/GRD seems to be a good commercial solution for those requiring that kind of support and wishing to have an advanced global scheduling system

Batch systems - external links

PBS Portable Batch System - http://www.mrj.com

LSF Load Sharing Facility - http://www.platform.com

LoadL Load Leveler - http://www.ibm.com

CODINE/GRD - http://www.gridware.com

Condor - ftp://ftp.cs.wisc.edu/condor

DQS - ftp://ftp.scri.fsu.edu/pub/DQS