

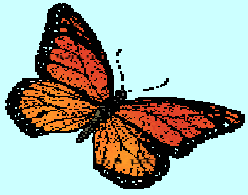
MONARC



Models Of Networked Analysis at Regional Centres Distributed System Simulation



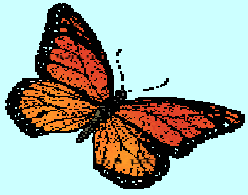
Iosif C. Legrand (CERN/CIT)
LCB, June 22, 1999



The GOALS of the Simulation Program



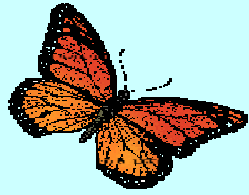
- ◆ To perform realistic simulation and modelling of the distributed computing systems, customised for specific HEP applications.
- ◆ To reliably model the behaviour of the computing facilities and networks, using specific applications software (OODB model) and the usage patterns.
- ◆ To offer a dynamic and flexible simulation environment.
- ◆ To provide a design framework to evaluate the performance of a range of possible computer systems, as measured by their ability to provide the physicists with the requested data in the required time, and to optimise the cost.
- ◆ To narrow down a region in this parameter space in which viable models can be chosen by any of the LHC-era experiments.



Design Considerations of the Simulation Program



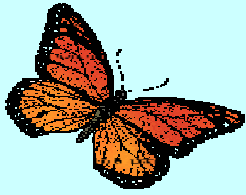
- ➔ The simulation & modelling task for the MONARC project requires to describe complex programs running in a distributed architecture.
- ✪ *Selecting Tools which allow one easily to map the logical model into the simulation environment.*
- ➔ A process oriented approach for discrete event simulation is well suited to describe concurrent running programs.
 - * **“Active objects”** (having an execution thread, a program counter, stack...) allow an easy way to map the structure of a set of distributed running programs into the simulation environment.



Design Considerations of the Simulation Program (2)



- ◆ This simulation project is based on **JavaTM** technology which provides adequate tools for developing a flexible and distributed process oriented simulation. Java has build-in **multi-thread** support for concurrent processing, which can be used for simulation purposes by providing a dedicated scheduling mechanism.
- ◆ The **distributed objects** support (through RMI or CORBA) can be used on distributed simulations, or for an environment in which parts of the system are simulated and interfaced through such a mechanism with other parts which actually are running the real application. The distributed object model can also provide the environment to be used for autonomous mobile agents.



Simulation Model



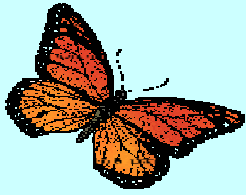
- ◆ It is necessary to abstract from the real system all components and their time dependent interaction.
- ◆ **THE MODEL** has to be equivalent to the simulated system in all important respects.

CATEGORIES OF SIMULATION MODELS

- ◆ Continuous time → usually solved by sets of differential equations
- ◆ Discrete time → Systems which are considered only at selected moments in time
- ◆ Continuous time + discrete event

Discrete event simulations (DES)

- ◆ **EVENT ORIENTED**
- ◆ **PROCESS ORIENTED**



Simulation Model(2)



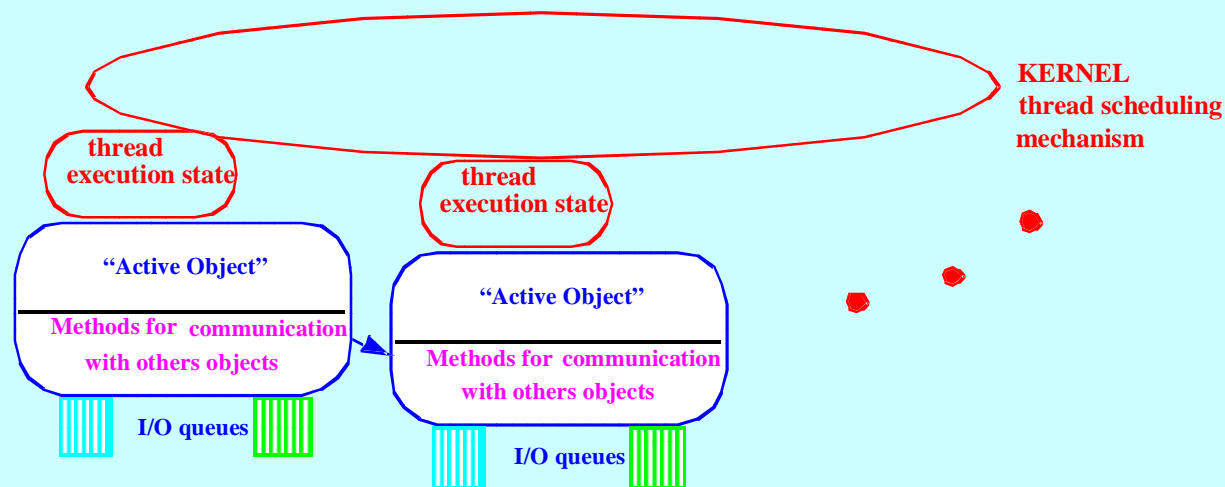
Process oriented DES Based on “ACTIVE OBJECTS”

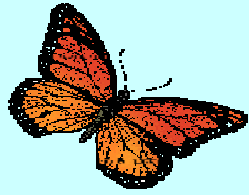
Thread: execution of a piece of code that occurs independently of and possibly concurrently with another one

Execution state: set of state information needed to permit concurrent execution

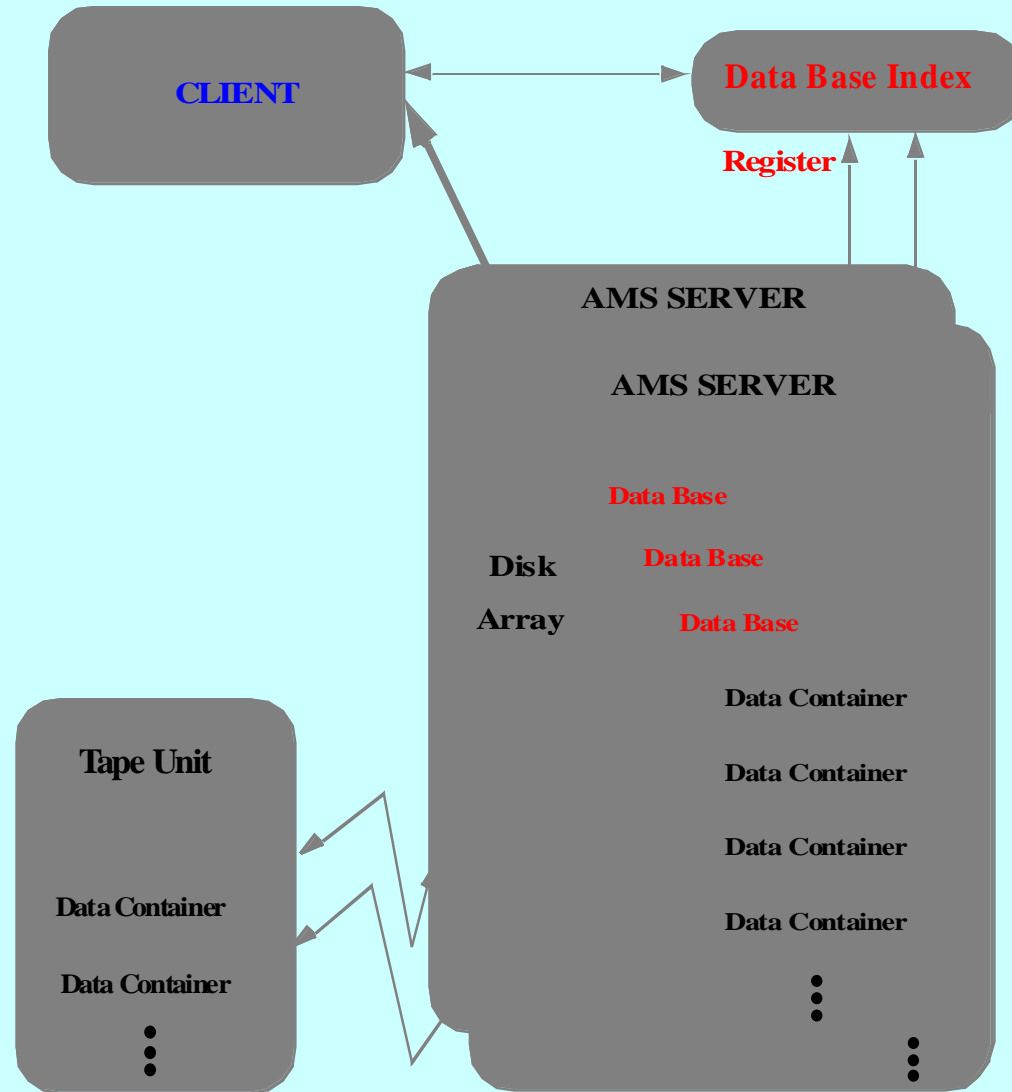
Mutual exclusion: mechanism that allow an action to performed on an object without interruption

Asynchronous interaction: signals/ semaphores for interrupts



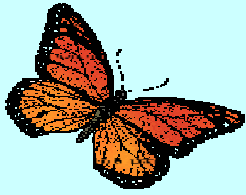


Data Model



It provides:

- ◆ Realistic mapping for an Object Database
- ◆ Specific HEP data structure
- ◆ Transparent access to any data
- ◆ Automatic storage management
- ◆ An efficient way to handle very large number of Objects.
- ◆ Emulate clustering factors for different types of access patterns.
- ◆ Handles related objects in different data bases.



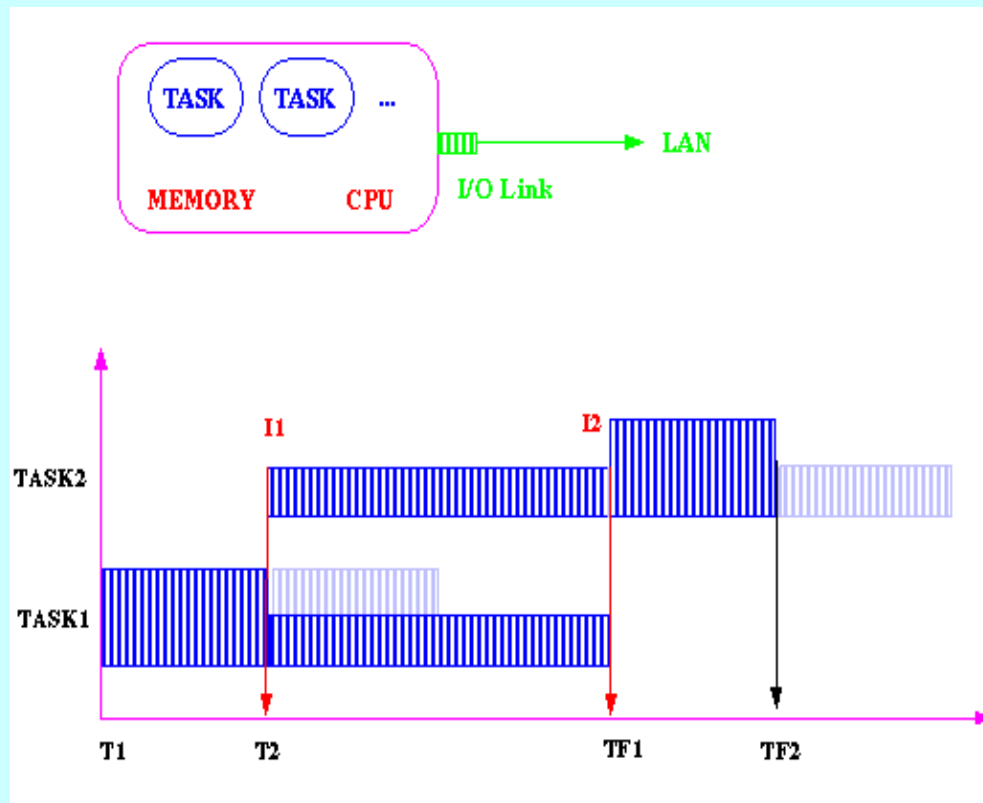
Multitasking Processing Model



Concurrent running tasks share resources (CPU, memory, I/O)

“Interrupt” driven scheme:

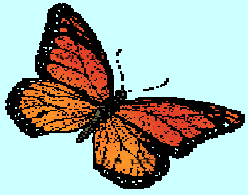
For each new task or when one task is finished, an interrupt is generated and all “processing times” are recomputed.



It provides:

An efficient mechanism to simulate multitask processing

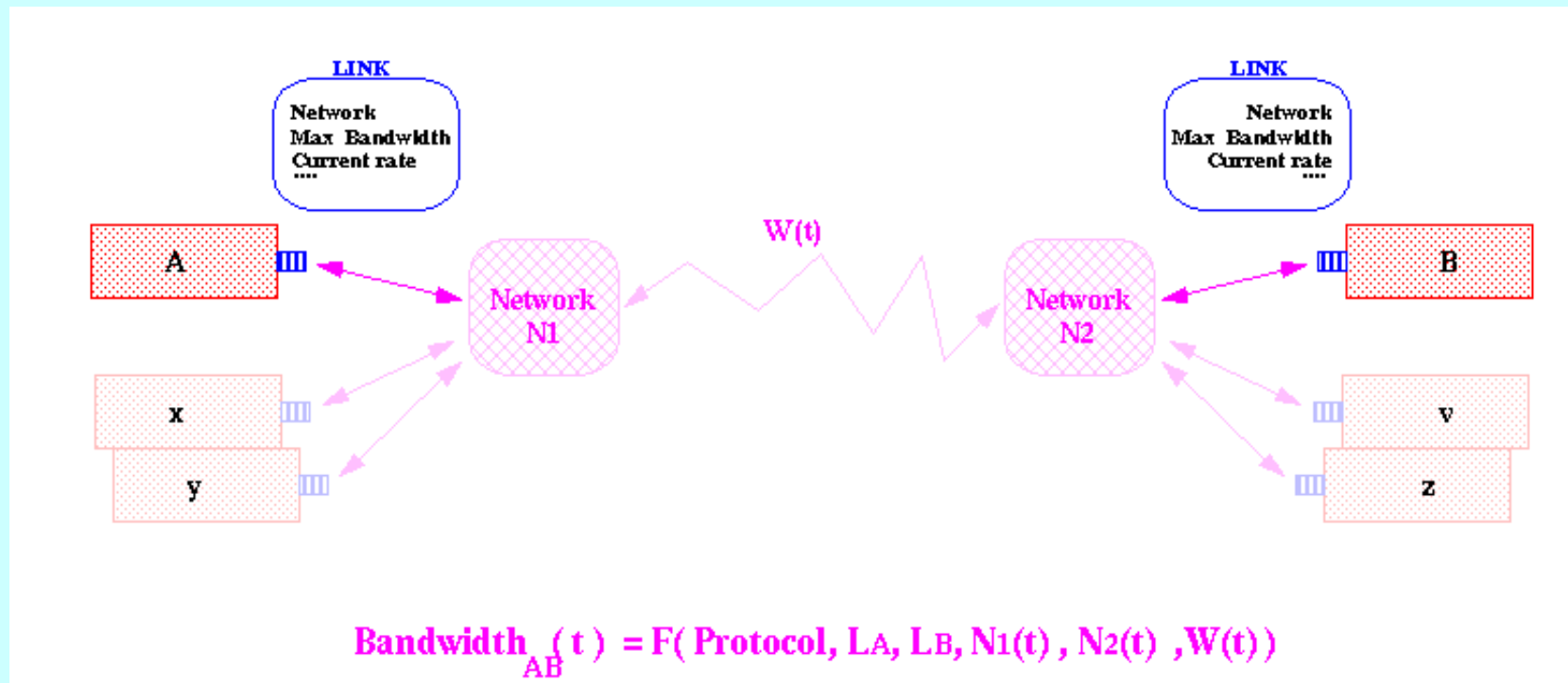
An easy way to apply different load balancing schemes



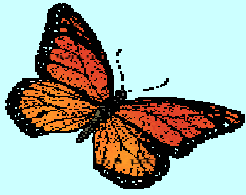
LAN/WAN Simulation Model



“Interrupt” driven simulation → for each new message an interrupt is created and for all the active transfers the speed and the estimated time to complete the transfer are recalculated.



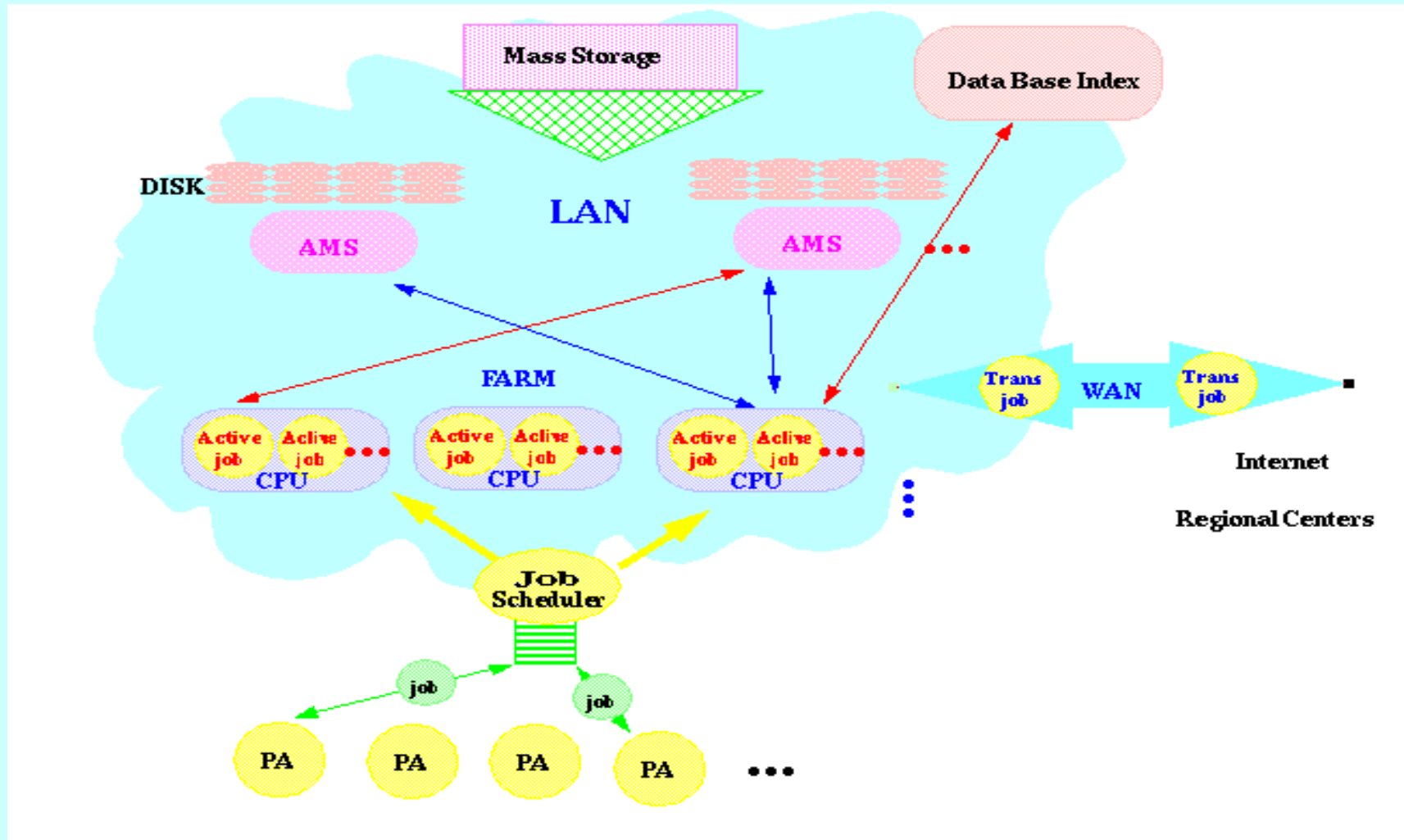
An efficient & realistic way to simulate concurrent transfers having different sizes / protocols.

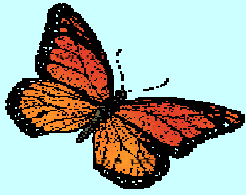


Regional Centre Model



Complex Composite Object



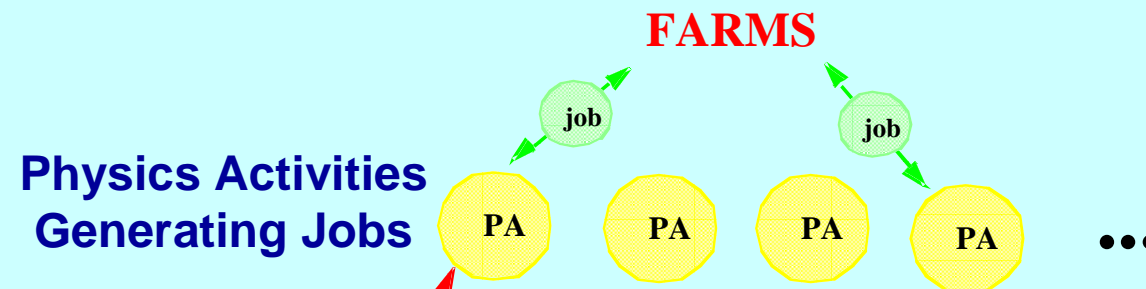


Arrival Patterns



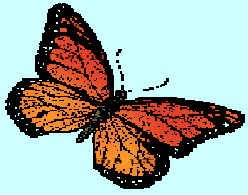
A flexible mechanism to define the Stochastic process of data processing

Dynamic loading of “Activity” Tasks, which are threaded objects and controlled by the simulation scheduling mechanism

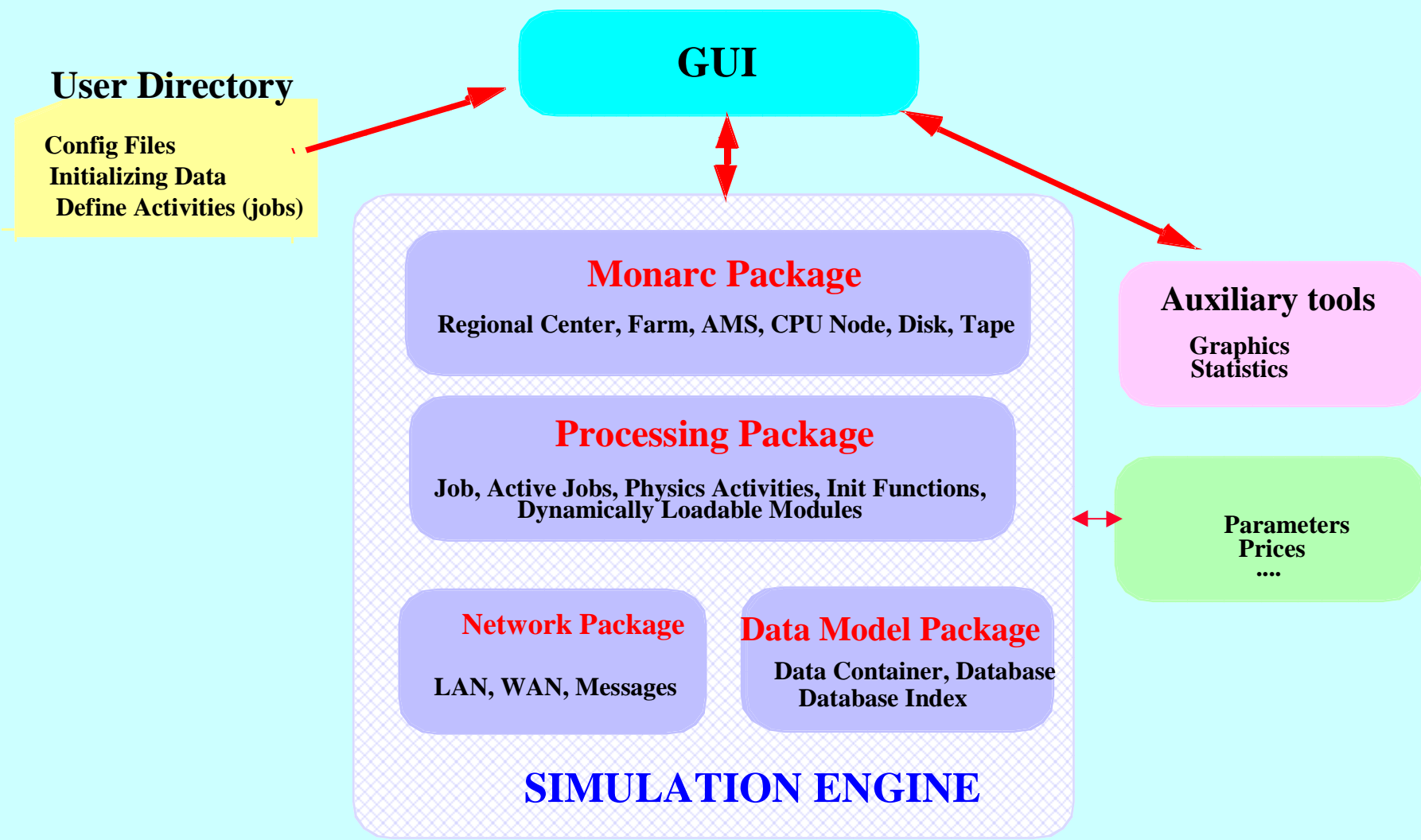


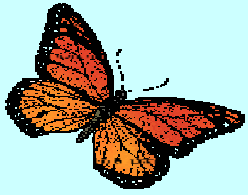
Each “Activity” thread generates data processing Jobs

```
for( int k=0; k< jobs_per_group; k++) {  
    Job job = new Job( this, Job.ANALYSIS, "TAG"+rc_name, event_offset, event_offset+events_to_process-1, null, null );  
    job.setAnalyzeFactors( 0.01, 0.005 );  
    farm.addJob( job);  
    sim_hold(1000.0 );  
}
```



The Structure of the Simulation Program





Input Parameters for the Simulation Program



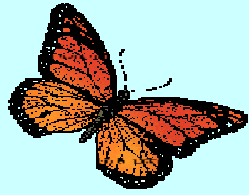
Correctly identify and describe the time response functions for all active components in the system. This should be done using realistic measurements.

The simulation frame allows one to introduce any time dependent response function for the interacting components.

$$\delta(\mathbf{T}_i) = \mathbf{F}(\delta(\mathbf{T}_{i-1}), \{\mathbf{SysP}\}, \{\mathbf{ReqP}\})$$

Response functions are based on “the previous state” of the component, a set of system related parameters (SysP) and parameters for a specific request (ReqP).

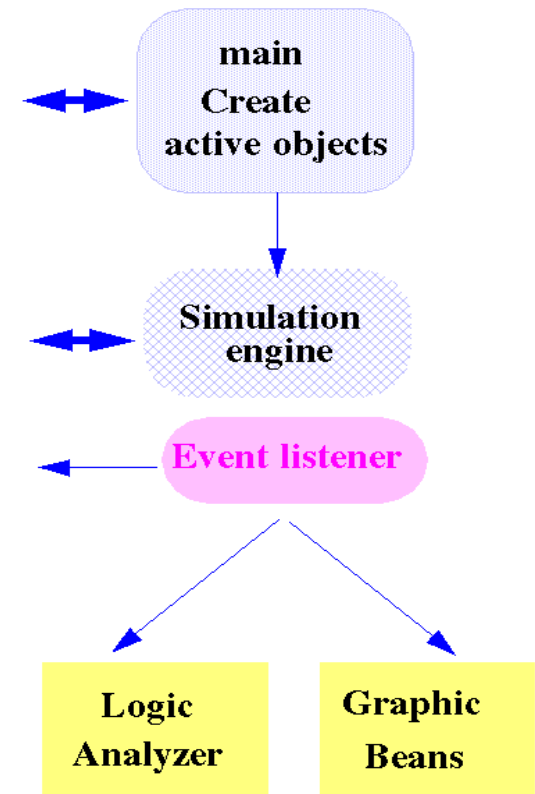
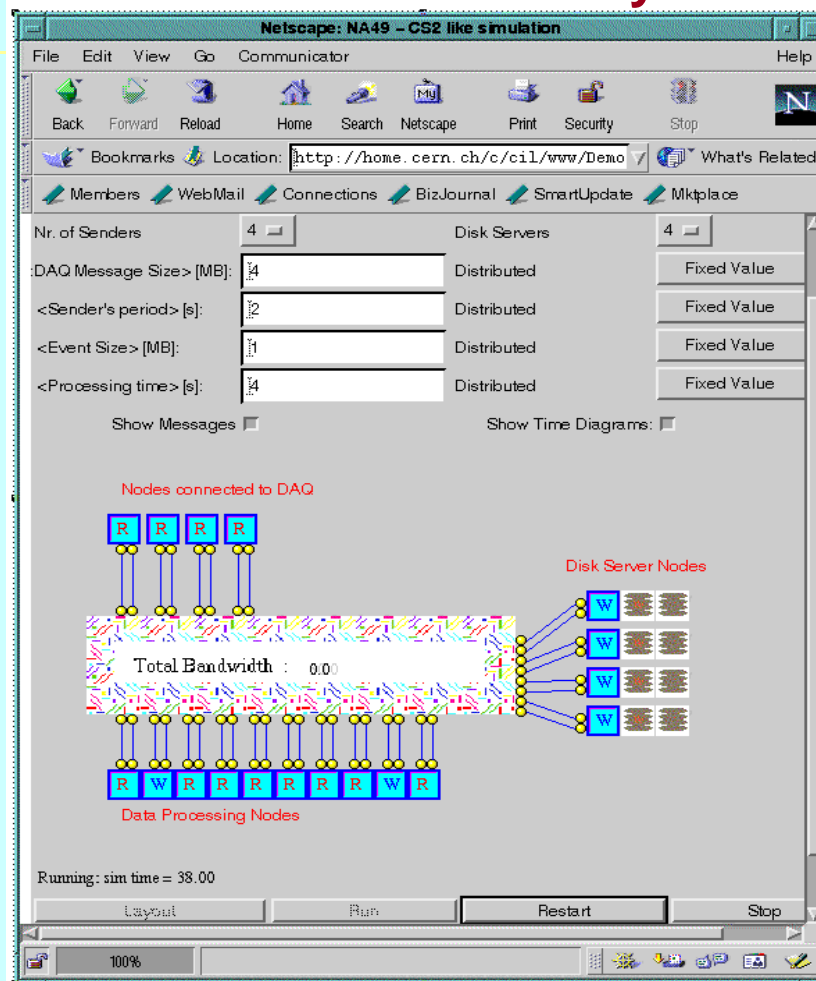
Allows to describe correctly **Highly Nonlinear Processes** or **“Chaotic” Systems** behavior (typical for caching, swapping...)



WEB GUI



CS-2 like Distributed System





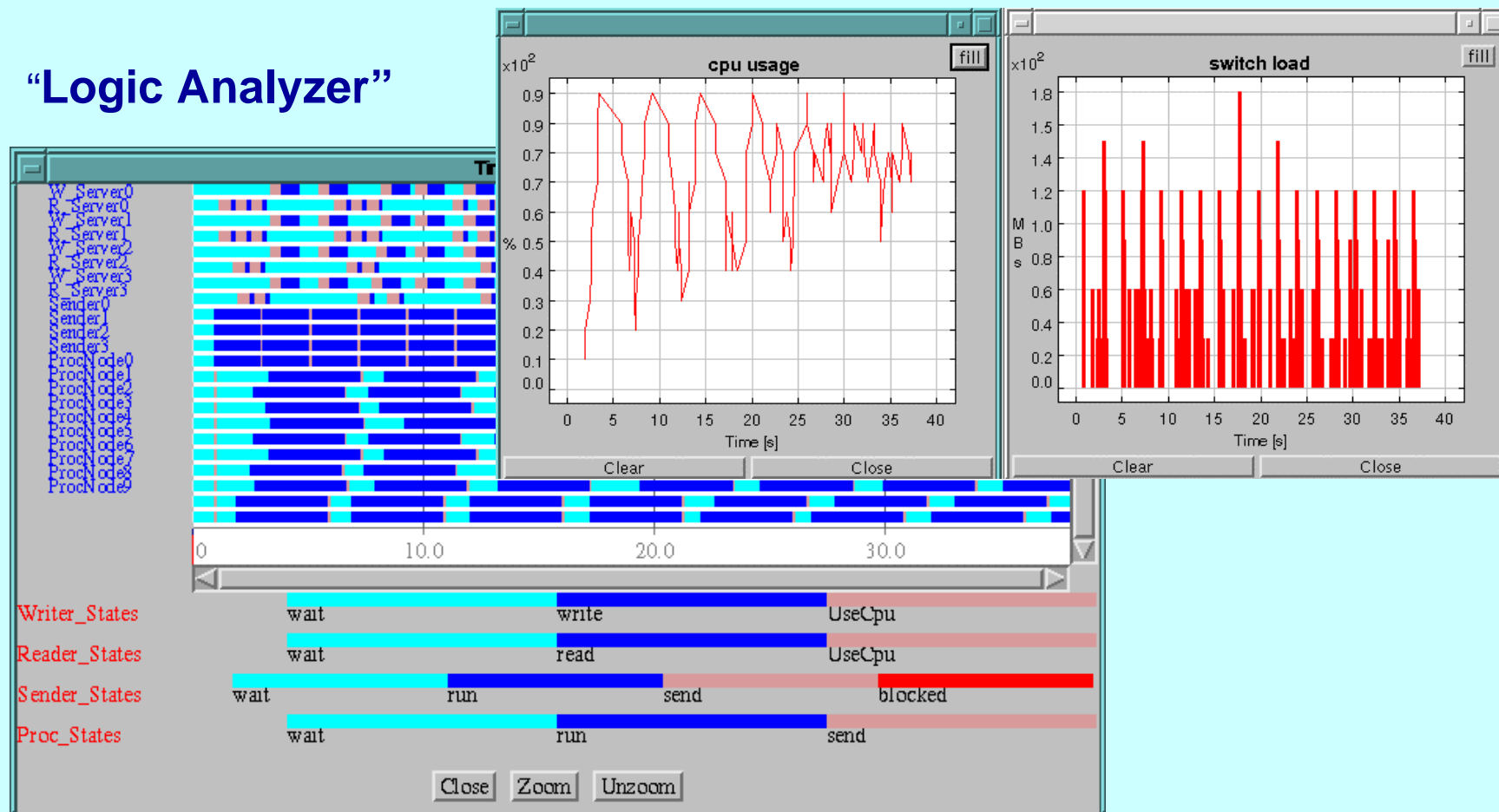
WEB GUI (2)

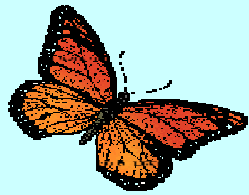


Graphic “Beans”

Monitoring different parameters

“Logic Analyzer”





Simulation GUI



Global Parameters

Parameter	Value	Units	Distributed
Proc_Time_RAW	250	[S195*s]	Fixed Value
Proc_Time_ESD	0.25	[S195*s]	Fixed Value
Proc_Time_AOD	2.5	[S195*s]	Fixed Value
Proc_Time_TAG	1.0	[S195*s]	Fixed Value
Analyze_Time_AOD	1.0	[S195*s]	Fixed Value
Analyze_Time_ESD	10.0	[S195*s]	Fixed Value
Analyze_Time_RAW	10.0	[S195*s]	Neg Exponential
Mem_Job_RAW_Proc	200.0	[MB]	Fixed Value

Input

Please enter RC name

 Add Remove OK Cancel

Price Evaluation

Item	Price	Comments
CPU power [S195]	400	
Memory [MB]	5	
Disk [GB]	10	
Link Port 1 MB/s	100	
Link Port 10 MB/s	300	
Link Port 15 MB/s	400	
Link Port 25 MB/s	600	
Link Port 100 MB/s	800	

cern Parameters

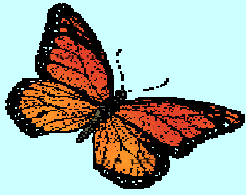
Parameter	Value	Comments
AMS_Servers	12	Nr. of AMS Servers
AMS_Link_Speed	20	I/O Bandwidth per ...
AMS_Disk_Size	5000	Disk Space per AMS ...
Process_Nodes	100	Nr. of Processing No...
Cpu_per_Node	50	Prossesing power [S...
Memory_per_N...	512	[MB]
Node_Link_Speed	10	[MB/s]
Max_Runnig_Jobs	500	The max nr. of sim...
MassStorage_Size	50	TB
MassStorage_Li...	20	[MB/s]
AMS_read_speed	15	[MB/s]

Init AMS function:
 Bandwidth Evaluation:

SHOW: Statistics, CPU, Local Data Traffic, Internet Data Traffic, Jobs, Efficiency

Buttons: Apply, Close

One may dynamically change parameters to control the simulation program

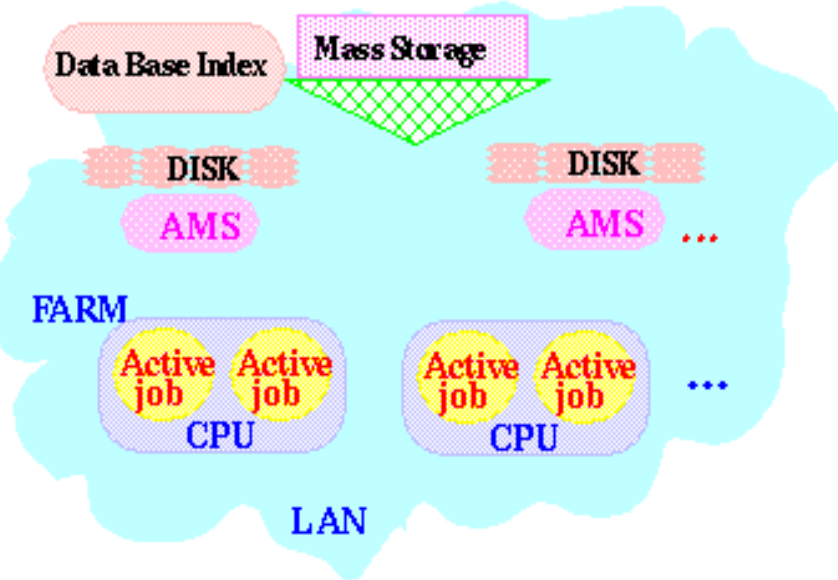


Example: Reconstruction



- ➔ It is performed in one Regional Centre
- ➔ Requires access to large amounts of data and a lot of processing power

Reconstruction Example



Tape may be necessary

Several AMS servers
Independent server for
reading data and writing results

Hundreds of CPU Nodes



Example: Reconstruction (2)



Parameters

Parameter	Value	Comments
AMS_Servers	10	Nr. of AMS Serve...
AMS_Link_Speed	10	I/O Bandwidth p...
AMS_Disk_Size	5000	Disk Space per A...
Process_Nodes	250	Nr. of Processing...
Cpu_per_Node	50	Prossesing powe...
Memory_per_Node	500	[MB]
Node_Link_Speed	10	[MB/s]
Max_Running_Jobs	500	The max nr. of ...
MassStorage_Size	50	TB
MassStorage_Link_Sp...	50	[MB/s]
AMS_read_speed	10	[MB/s]

Init AMS function: InitAMS_cern
Bandwidth Evaluation: Bandwidth_cern

SHOW

- Statistics
- CPU
- Local Data Traffic
- Internet Data Traffic
- Jobs
- Efficiency

Statistics @cern

Parameter	Value
Estimated Price [k\$]	5,703.00
Nr. of Jobs Processed	1000
Nr. of Jobs Aborted	0
CPU usage- Integrated mean [%]	50.647
Total CPU used [SI95*s 10^6]	2,500,000
AMS servers write [GB]	1,000.00
AMS servers read [GB]	10,000.00
Processed Events	1.0E7
Processing Rate [events/s]	25.311

cern-CPU & Memory

Integral :
cpu : 20,000,000.00
mem : 23,191,690.92

cern-Local Network

Integral :
ams : 21,000,000.00
cpu : 11,000,000.00
tape : 10,000,000.0

cern - CPU & Memory Graph

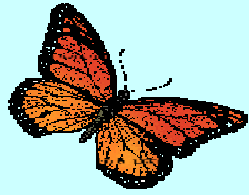
Y-axis: % (x10²)
X-axis: Time [s] (x10⁵)

Legend: cpu (red), mem (blue)

cern - Local Network Traffic Graph

Y-axis: MB / s (x10²)
X-axis: Time [s] (x10⁵)

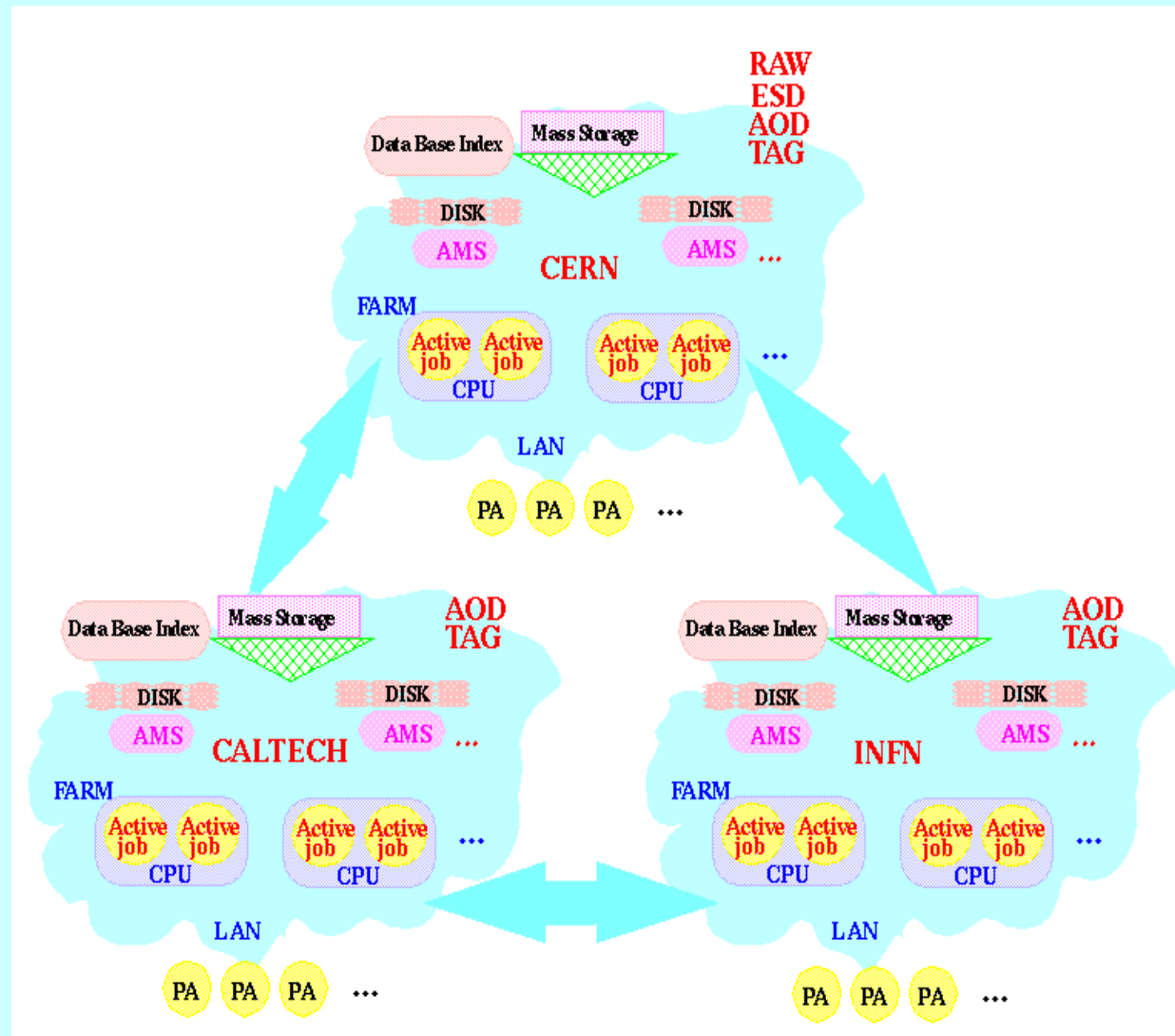
Legend: ams (red), cpu (blue), tape (green)

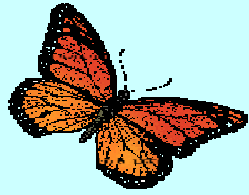


Example : Physics Analysis

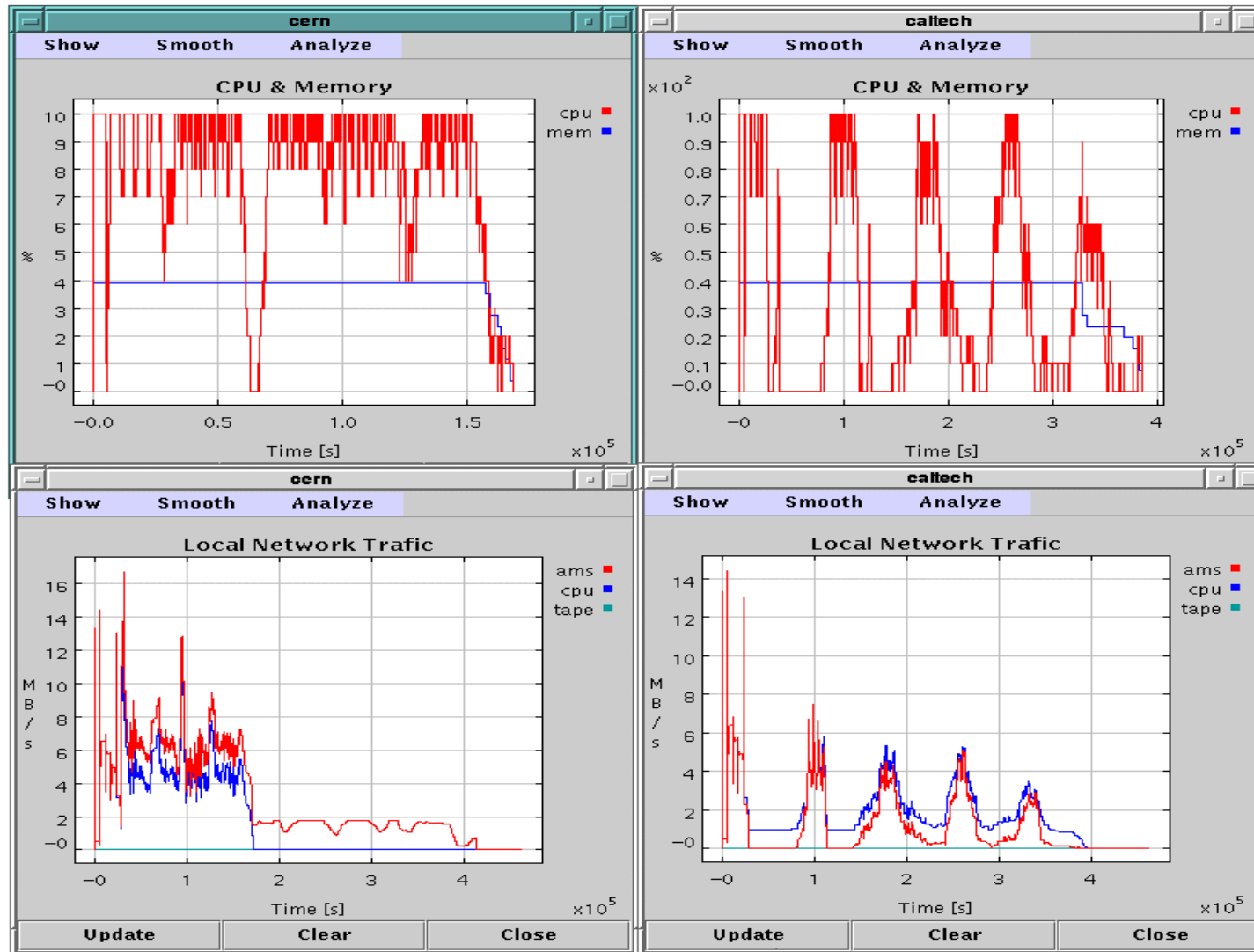


- ➔ Similar data processing jobs are performed in several RCs
- ➔ Each Centre has “TAG” and “AOD” databases replicated.
- ➔ Main Centre provides “ESD” and “RAW” data
- ➔ Each job processes AOD data, and also a fraction of ESD and RAW





Example: Physics Analysis (2)



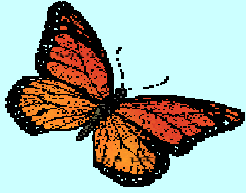


Example: Physics Analysis (3)



The screenshot displays the Monarc Simulation software interface, divided into several windows:

- Monarc Simulation (Regional Centers):** Lists 'cern', 'caltech', and 'infn'. It includes buttons for 'Global Parameters', 'Estimated Prices', 'Add', and 'Remove'. A status bar shows 'Finished ! Restart Simulation' and 'Exit'. The time is 'd:5 h:8'.
- Parameters (cern):** A table of simulation parameters with columns for Parameter, Value, and Comments. Below the table are fields for 'Init AMS function' and 'Bandwidth Evaluation', and buttons for 'Apply' and 'Close'.
- Internet Traffic (cern):** A graph showing traffic in MB/s over time (x10⁵ s). The legend indicates 'caltech' (red) and 'infn' (blue). A pop-up window shows the integral values: 'caltech : 300,055.0' and 'infn : 300,055.0'.
- Internet Traffic (caltech):** A similar graph for the caltech center, showing only the red line for 'cern'.



Summary



A CPU- and code-efficient approach for the simulation of distributed systems has been developed for MONARC

- provides an easy way to map the distributed data processing, transport, and analysis tasks onto the simulation
- can handle dynamically any Model configuration, including very elaborate ones with hundreds of interacting complex Objects
- can run on real distributed computer systems, and may interact with real components
- * The Java (JDK 1.2) environment is well-suited for developing a flexible and distributed process oriented simulation.
- * This Simulation program is still under development, and dedicated measurements to evaluate realistic parameters and “validate” the simulation program are in progress.