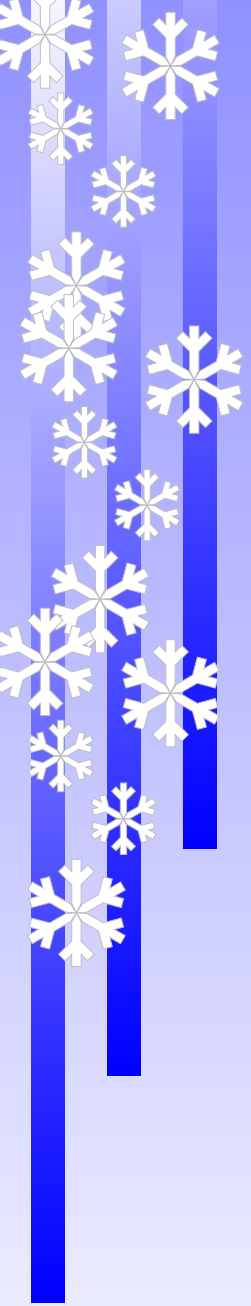


Autonomic Web-based Simulation

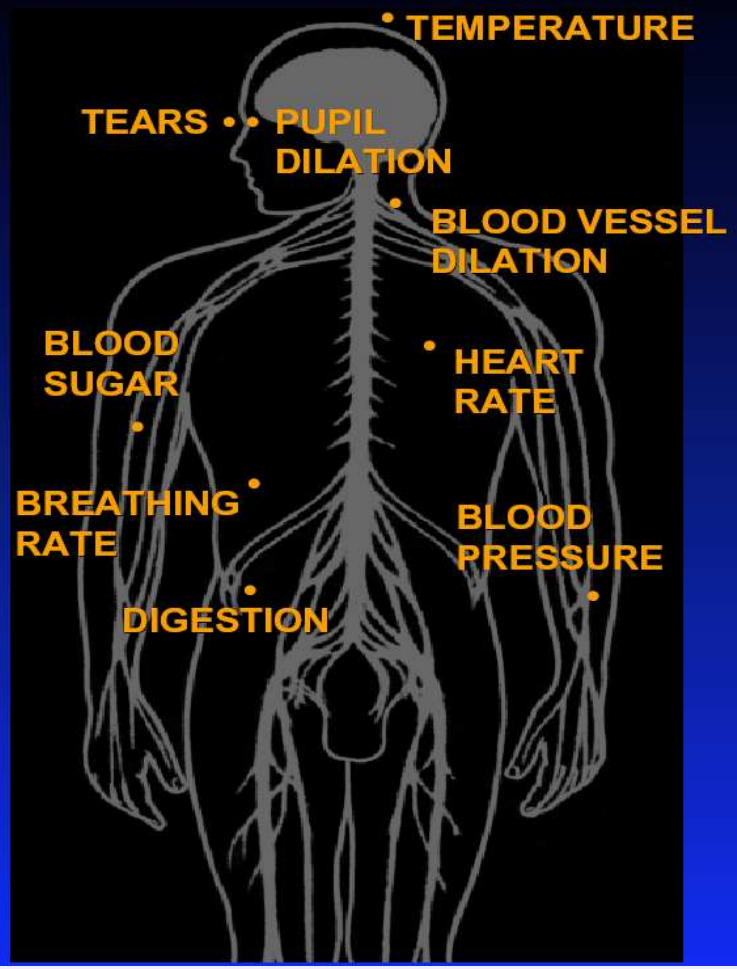
Yingping Huang and Gregory Madey
Computer Science and Engineering
University of Notre Dame

Autonomic Web-based Simulation

- Autonomic Web-based Simulation =
  - Web-based Simulation +
  - Autonomic Computing
- Motivations
  - Many scientific simulations are large programs which despite careful debugging and testing will probably contain errors when deployed to the Web for use
  - Developers of large-scale web-based simulations have experienced increased complexity in their software systems due to the complex integration of different pieces of services.
- Goal
  - Self-manageable Web-based simulations

Human Nervous System

The Autonomic Nervous System Monitors and Regulates:



Autonomic Computing Vision

Autonomic Computing Vision

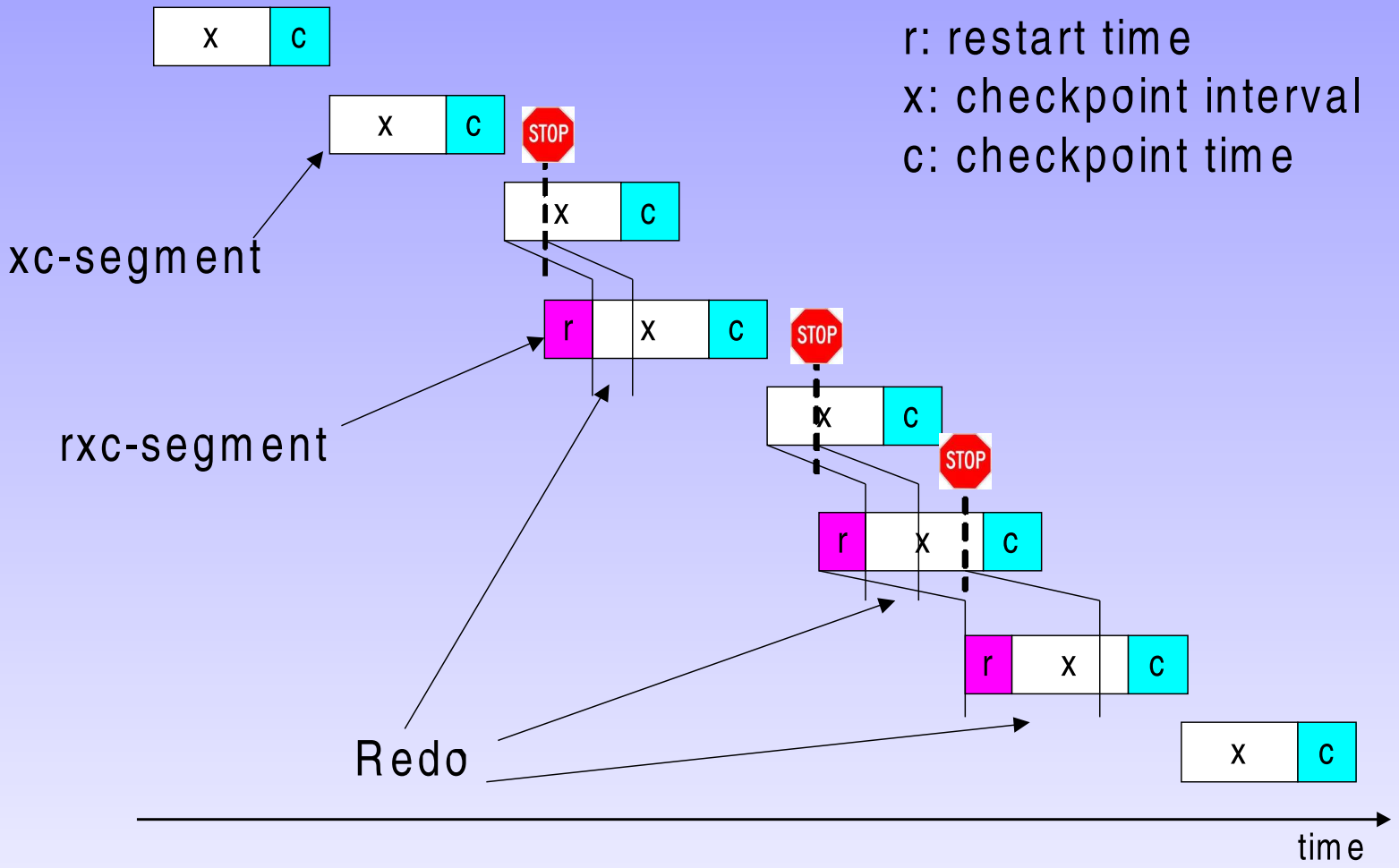
AWS Requirements

1. Simulation checkpointing and restarting
2. Simulation self-awareness and proactive failure detection
3. Self-manageable computing infrastructure to host simulations

Ckpt 4 Self-healing/optimizing

- ✓ Checkpointing is used in simulations, databases, systems, and operations research
- ✓ Determining optimal checkpoint interval is not trivial
  - ⋆ Excessive checkpointing results in performance degradation $\Longrightarrow$ longer execution time
  - ⋆ Deficient checkpointing yields expensive redo $\Longrightarrow$ longer execution time
- ✓ An optimization problem is formed

Modeling Simulation Execution



Expected Execution Time

- ✓ T_{total}: Expected total execution time is the sum of the following 4:
  - ⋆ T_{work}: Time to complete all computations with the assumption of no checkpointing and no failure
  - ⋆ $T_{checkpoint}$: Time to write checkpoint data to files or database
  - ⋆ $T_{restart}$: Time to detect failures and restore data from last checkpoints
  - ⋆ T_{redo}: Time to redo computations to the points of failures

Assumptions for Analytical Models

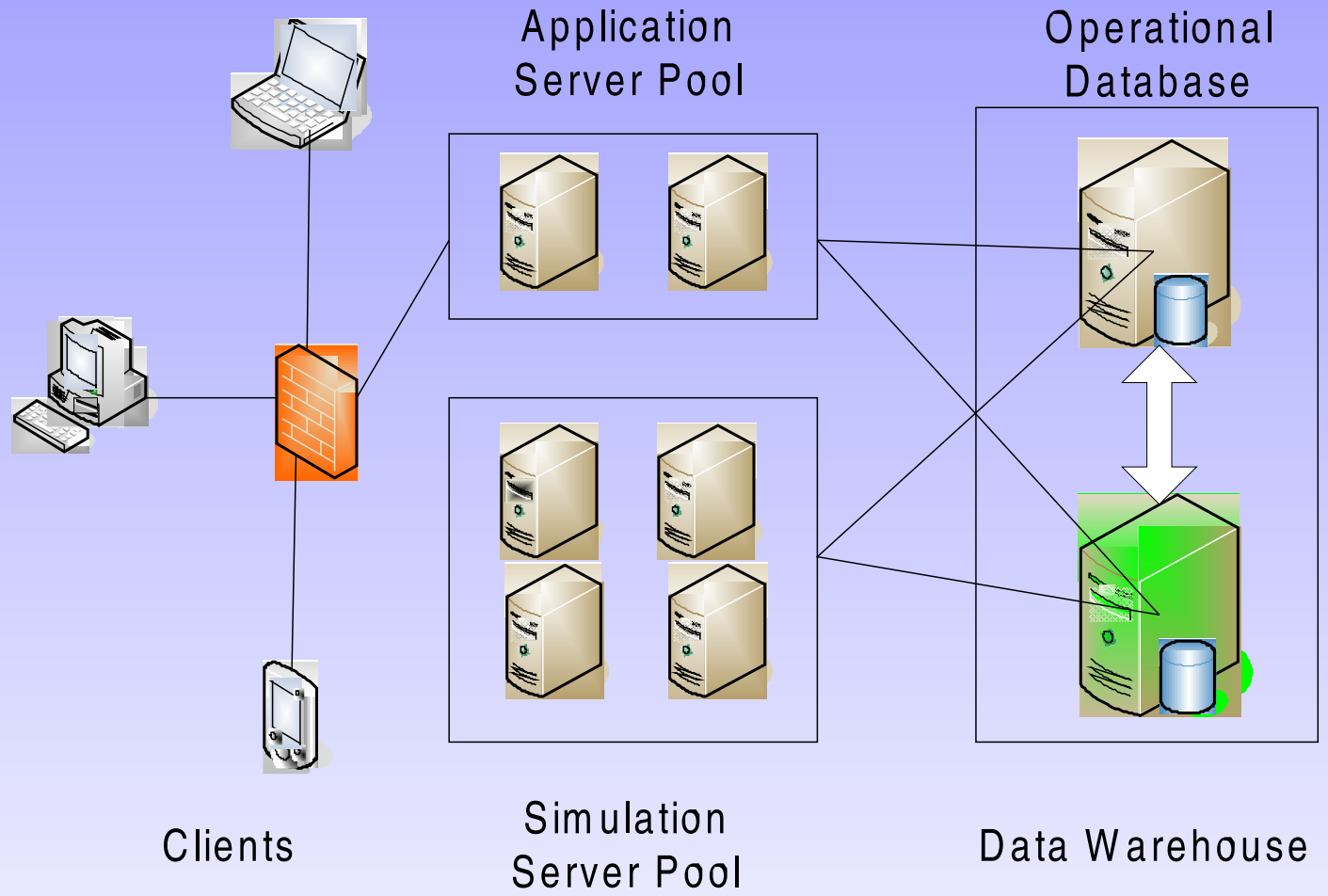
- ✓ Assumptions:
  - ⋆ $MTTF = M$ where M is a constant. Failures occur according to a Poisson process with arrival rate $\frac{1}{M}$. $\Longrightarrow$
    - → The probability to complete t time units without failure is $p(t) = e^{-\frac{t}{M}}$
    - → The probability distribution function is $\frac{1}{M}e^{-\frac{t}{M}}$
  - ⋆ For an execution segment, checkpoint time is c and restart time is r (if it's an rxc-segment), where c and r are constants
- ✓ Critical to determine
  - ⋆ Fraction of redo over an execution segment
  - ⋆ The expected number of failures

Requirement 2: J2SE 5.0

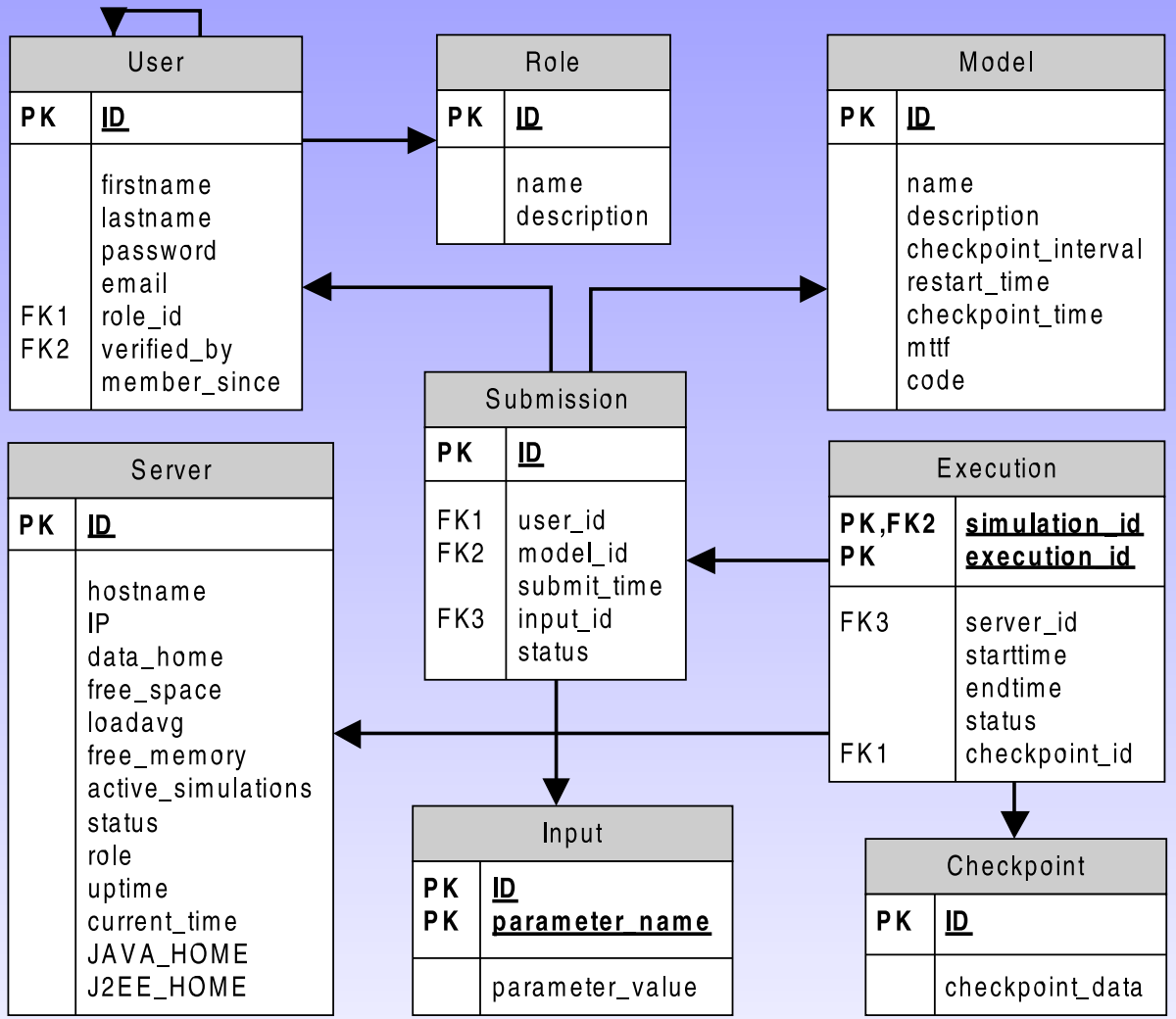
- The information exposed by the monitoring and management APIs in J2SE 5.0 can be used in:
  - External monitoring and management using external monitoring software
  - Internal monitoring and management by adding logic inside simulation

Managed Resource	*Interfaces in java.lang.management*
Memory	MemoryMXBean MemoryPoolMXBean MemoryManagementMXBean
	RuntimeMXBean
	GarbageCollectorMXBean
CPU	OperatingSystemMXBean ThreadMXBean
	RuntimeMXBean

Req 3: Self-* Infrastructure



Data Model 4 Self-awareness



Self-configuring

- Self-configuring involves autonomatic incorporation of new components and autonomic component adjustments to new conditions
- Self-configuring tasks
  - Self-configuring web interface
  - Self-configuring firewall/router
  - Self-configuring simulation servers
  - Self-configuring application server

Self-configuring Web Interface

- ✓ Frequent database schema changing due to research uncertainty yields corresponding of web interface.
- ✓ Web interface can be changed automatically with multi-record format

ID	iterations	density	temperature	pH
1	100	.1	100	7.1
2	200	.2	200	7.2
3	300	.3	300	7.3
4	400	.4	400	7.4

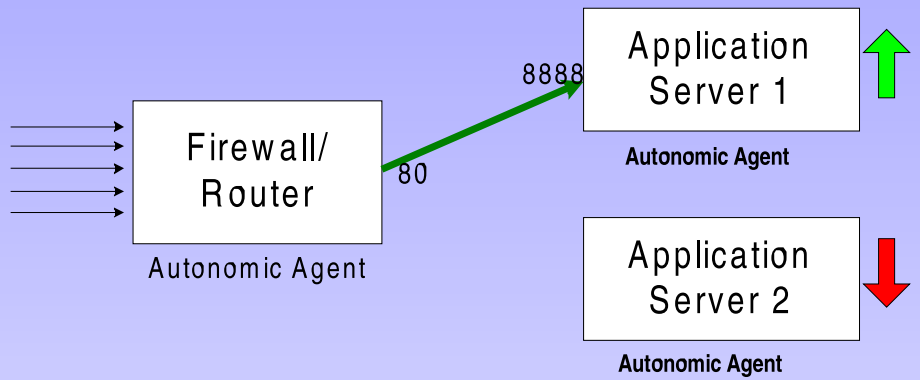
Single Record Format

ID	Parameter name	Parameter value
1	iterations	100
1	density	.1
1	temperature	100
1	pH	7.1
2	iterations	200
2	density	.2
...	...	...

Multi Record Format

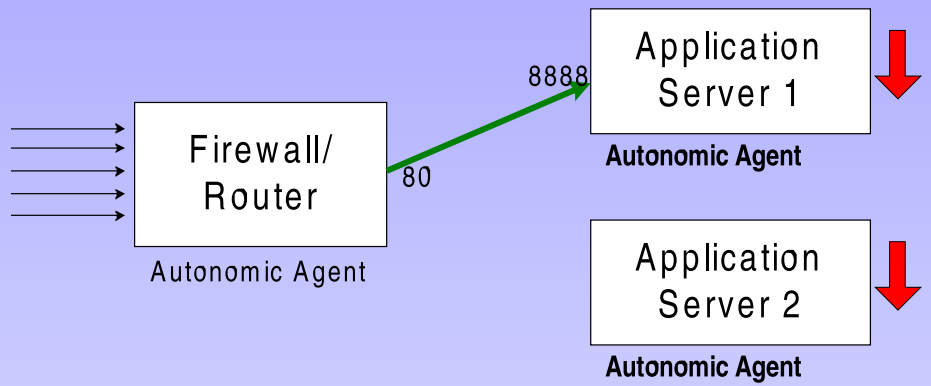
Self-configuring Firewall/Router

- IP is forwarded to application server 1



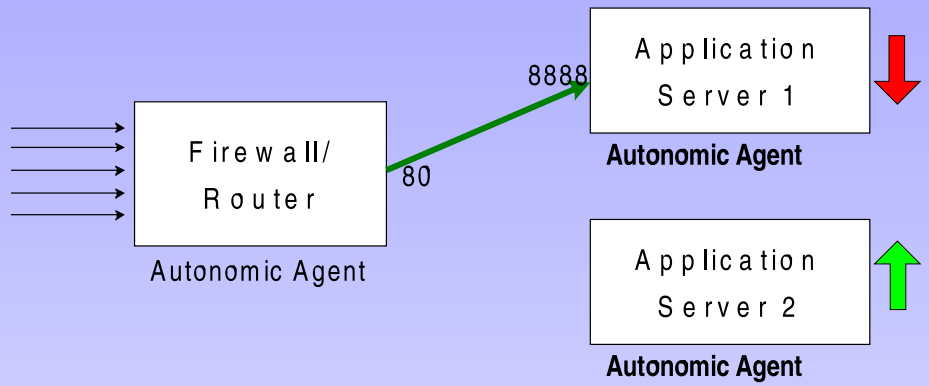
Self-configuring Firewall/Router

- ✓ IP is forwarded to application server 1
- ✓ Failure of application server 1 is detected



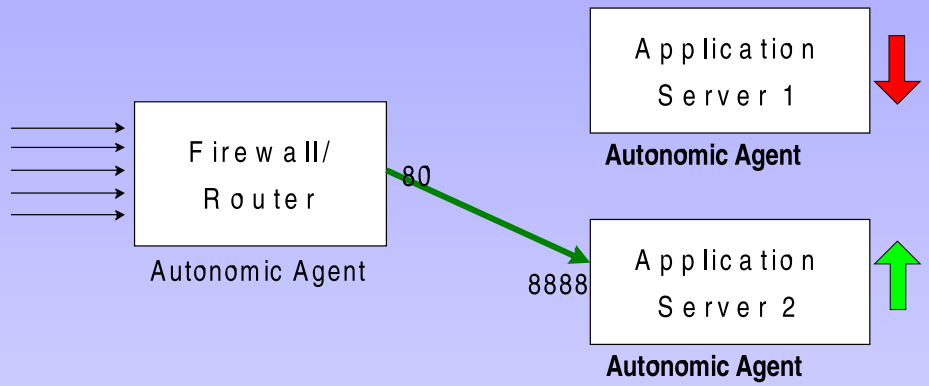
Self-configuring Firewall/Router

- √ IP is forwarded to application server 1
- √ Failure of application server 1 is detected
- √ Local autonomic agent starts application server 2



Self-configuring Firewall/Router

- IP is forwarded to application server 1
- Failure of application server 1 is detected
- Local autonomic agent starts application server 2
- IP is forwarded to application server 2



Self-configuring Simulation Servers

- ✓ Autonomic agents are running on simulation servers and new simulation servers are discovered by inserting records into the Server table
- ✓ Load metrics such as load average are updated every 5 seconds in the Server table
- ✓ Old records are inserted into Server_History by a database trigger, and are used for load balancing and simulation migration

Self-healing

- Self-healing can be accomplished by automatically detecting, diagnosing, and repairing localized software or hardware problems. Some sort of redundancy is necessary to achieve self-healing.

- Self-healing application servers

1. Detect application server failure by probing it using wget
2. Local agent starts another application server
3. Firewall/Router runs iptables command for IP forwarding

Self-healing

- Self-healing can be accomplished by automatically detecting, diagnosing, and repairing localized software or hardware problems. Some sort of redundancy is necessary to achieve self-healing.

- Self-healing application servers
- Self-healing simulation servers

1. Detect simulation server failure by timing out of autonomic agents
2. All simulations running on the simulation server are crashed
3. All crashed simulations are re-dispatched by the autonomic manager inside the database server

Self-healing

- Self-healing can be accomplished by automatically detecting, diagnosing, and repairing localized software or hardware problems. Some sort of redundancy is necessary to achieve self-healing.

- Self-healing application servers
- Self-healing simulation servers
- Self-healing running simulations

1. Failures are detected either by the Java Monitoring and Management APIs or timing out
2. Simulations are killed by local agents
3. Crashed simulations are re-dispatched by the autonomic manager inside the database server

Self-healing

- Self-healing can be accomplished by automatically detecting, diagnosing, and repairing localized software or hardware problems. Some sort of redundancy is necessary to achieve self-healing.

- Self-healing application servers
- Self-healing simulation servers
- Self-healing running simulations
- Self-healing database servers

1. Database server and listener are monitored by making periodical connections
2. Alert log is monitored for number of significant errors, especially ORA-00600 errors.
3. Tablespace capacity is monitored, so that it exceeds threshold, new space is allocated

Self-optimizing

- Self-optimizing involves automatic tuning of performance related parameters. The idea of global optimization is useful for self-optimizing. However, usually the performance related parameters cannot be changed dynamically without rebooting the services.
- Self-optimizing task
  - Self-optimizing simulation servers by load balancing and simulation migration
  - Self-optimizing simulations by using optimal checkpoint interval

Self-protecting

- Self-protecting means the system automatically defends against malicious attacks or cascading failures. It use early warnings to anticipate and prevent system wide failures.
- Access to the computing infrastructure is controlled through user roles.
- Self-protecting tasks
  - Firewall is configured to allow only port 80 open to public
  - Users must register and be verified by system administrators
  - Users are assigned roles: admin, normal and not
  - Early warning of OutOfMemoryError were used to anticipate failures

Conclusions

- ✓ The following contributions are reported:
  - ⋆ Derivation of mathematical models to calculate the optimal checkpoint interval and to predict expected total execution time
  - ⋆ Implementation of autonomic web-based simulation and its application to the NOM simulation

Guess What...

- ✓ This is not PowerPoint...
- ✓ This is done by Latex + Prosper

Thank You