System-level Virtualization for High Performance Computing

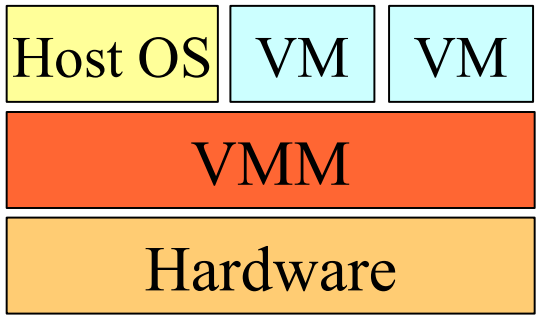
Geoffroy Vallee
Thomas Naughton
Christian Engelmann
Hong Ong
Stephen L. Scott

PDP – Toulouse, France – Feb 2008

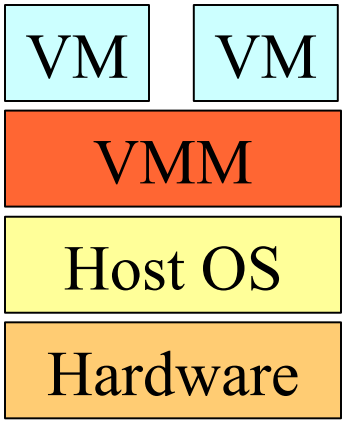
OAK RIDGE National Laboratory

Introduction to Virtualization

- **System-level virtualization studied since the 70's (Goldberg, Popek)**
- **Key concepts:**
  - ***Virtual Machine (VM), guest OS*: complete operating system running in a virtual environment**
  - ***Host OS*: operating system running on top of the hardware, interface between the user and the VMM and VMs**
  - ***Virtual Machine Monitor (VMM), Hypervisor*: manage VMs (scheduling, hardware access)**



Type-I Virtualization

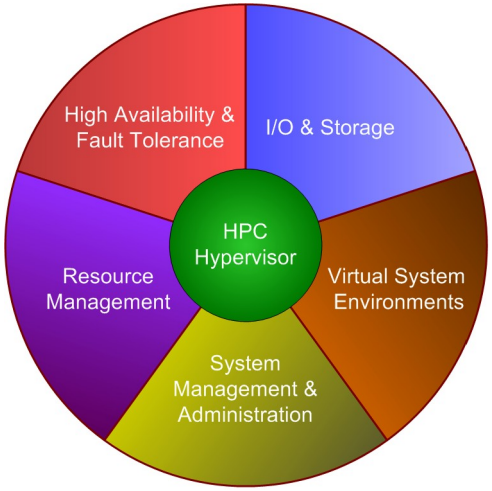


Type-II Virtualization

The HPC Context

- The system should not interfere with running application(s)
  - minimize the OS footprint/noise: Catamount, CNK
  - priority to the application for resources access
- Support large-scale systems
  - thousands of distributed components
  - fault tolerance
  - system partitioning: compute nodes vs. service nodes
- Support HPC applications, e.g., MPI applications

Hypervisor for HPC



- **What does it mean?**
  - *VMM for HPC*
    - minimize the VMM memory footprint
    - VMM/HostOS system footprint
      - only the HostOS can execute privileged instructions on the behalf of VMs
      - possible domain context switches
  - Provide a suitable *execution environment* to HPC applications
  - *Fault tolerance*
  - *System Management* of VMs/VMMs/HostOSes in large-scale distributed systems
  - *I/O & Storage*
    - efficient access to resources
    - resource sharing between VMs running on different nodes
- **Current virtualization solutions are not suitable for HPC**

Hypervisor for High-Performance Computing

VMM for High-Performance Computing

- Minimize the system footprint
  - reduce the default VMM/HostOS memory usage
  - use hardware optimizations (AMD nested pages, hardware IOMMU)
- Minimize the context switches between domains
  - pin Vms/HostOSes/VMM to a core/processor
- Avoid the usage of HostOSes for a direct access to the bare hardware
  - isolation vs. resource sharing
  - “VMM-bypass”
- Guarantee experiment reproductability
  - Have the same behavior between different application runs

Virtual System Environment

Virtual System Environment (VSE)

- Objective: *"Adapt the operating system to the application, instead of adapting the application to the OS"*
  - The science resides in the applications, not in the operating systems or run-times
  - Goal: application developers should not "port" their application every time they want to use a new execution platform
- System-level virtualization does offer interesting features
  - Application isolation within virtual machines (users can do whatever they want)
  - All standard UNIX tools can be used

Challenges for the Usage of VSE in Distributed Systems

- System management
  - How can we deploy & manage both the HostOS and the VMs?
  - How can users specify the system within a VM?
  - How can sysadmin specify their constraints regarding execution environments?
  - How can we switch between different virtualization solutions?
  - How can we switch between a virtual environment and a normal environment (*e.g.*, disk-less, disk-full)?

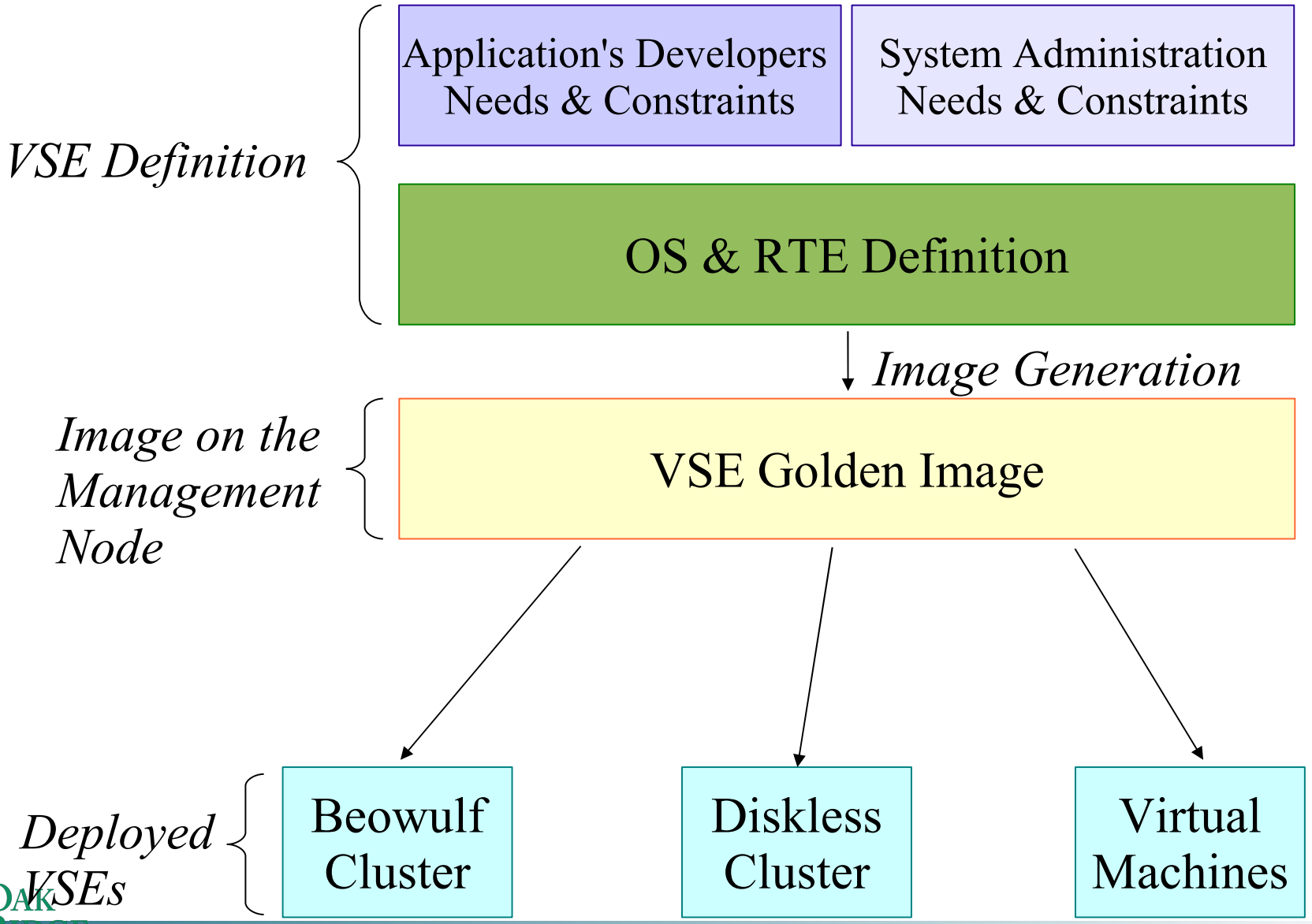
Important lack of tools

- Selected approach: the definition of *Virtual System Environments*

OSCAR-V: OSCAR Extension for Virtual Systems

- **Implement VSEs support**
  - without rewriting everything from scratch
  - potential support of all RPM and Debian based Linux distributions
  - abstracting existing system-level virtualization solutions
- **Provide an integrated solution for:**
  - the deployment & management of both HostOSes and VMs
  - the specification of VSE for both the user point-of-view and the sysadmin point-of-view
  - unique interface for the manage of VMs: concept of *profile*
  - possible switch between disk-less, beowulf and partitioned systems (ongoing work)

VSE Management - Overview

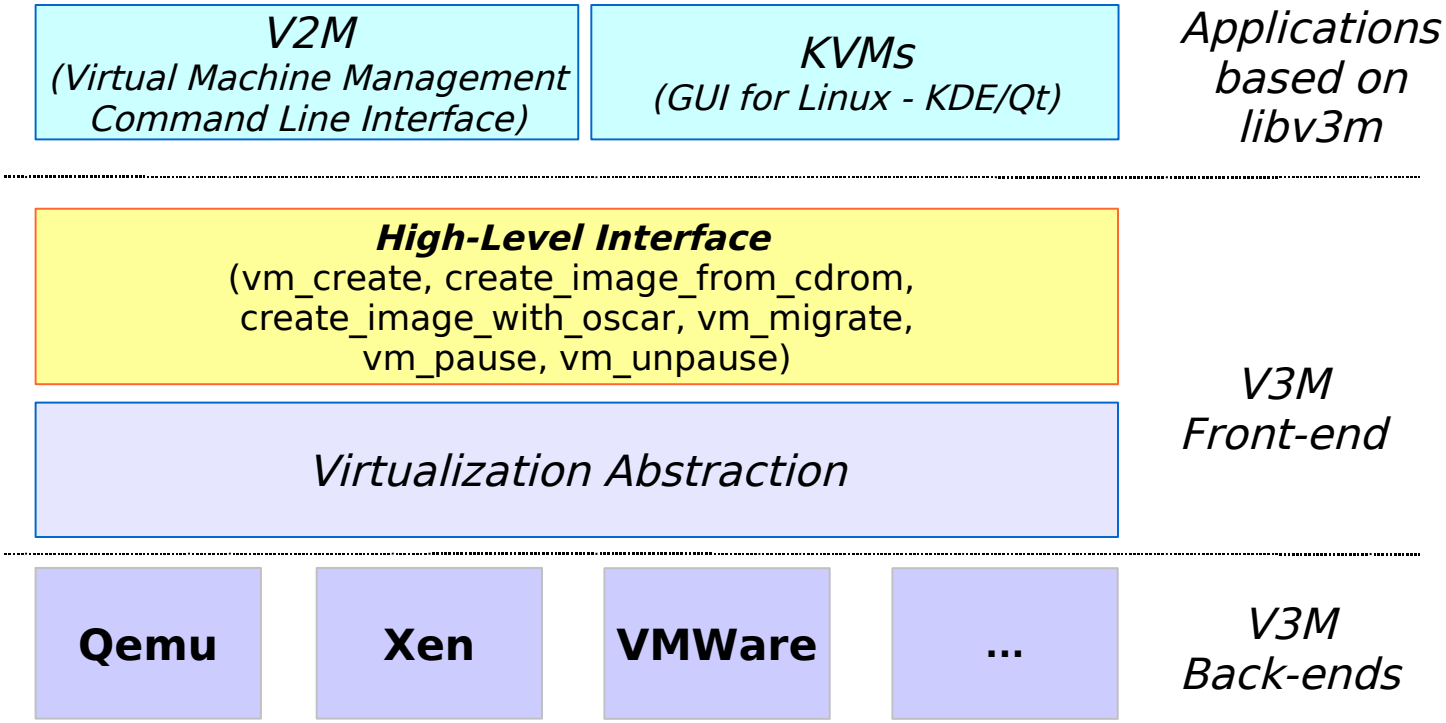


Abstraction of the System-Level Virtualization Solutions

- Users do not care about the technical details
  - hide all the configuration details: Virtual Machine Management - V2M
  - provide a simple API for the definition of VMs: *profiles*
- Profile example

```
<?xml version="1.0"?>
<!DOCTYPE profile PUBLIC "" "v3m_profile.dtd">
<profile>
        <name>test</name>
        <type>Xen</type>
        <image size="50">/home/gvallee/temp/v2m/test_xen.img</image>
        <nic1>
                <type>TUN/TAP</type>
                <mac>00:02:03:04:05:06</mac>
        </nic1>
</profile>
```

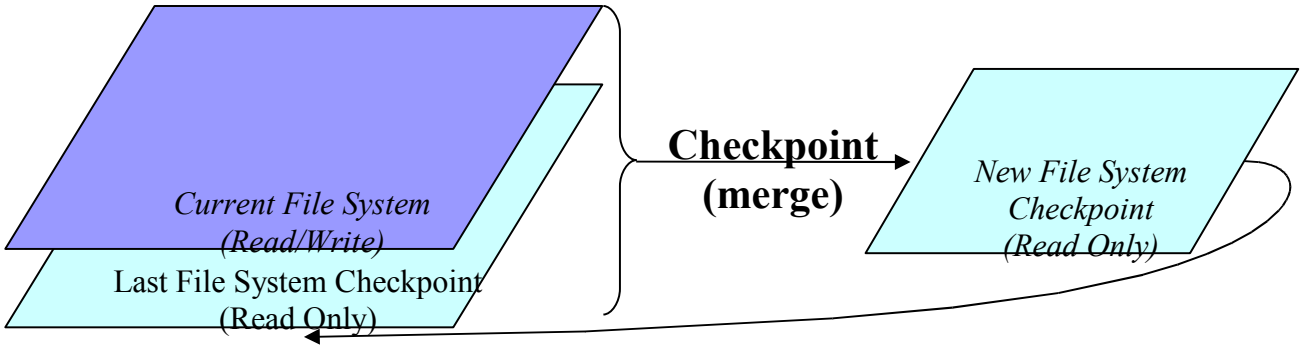
V2M - Architecture



System-level Virtualization and High Availability

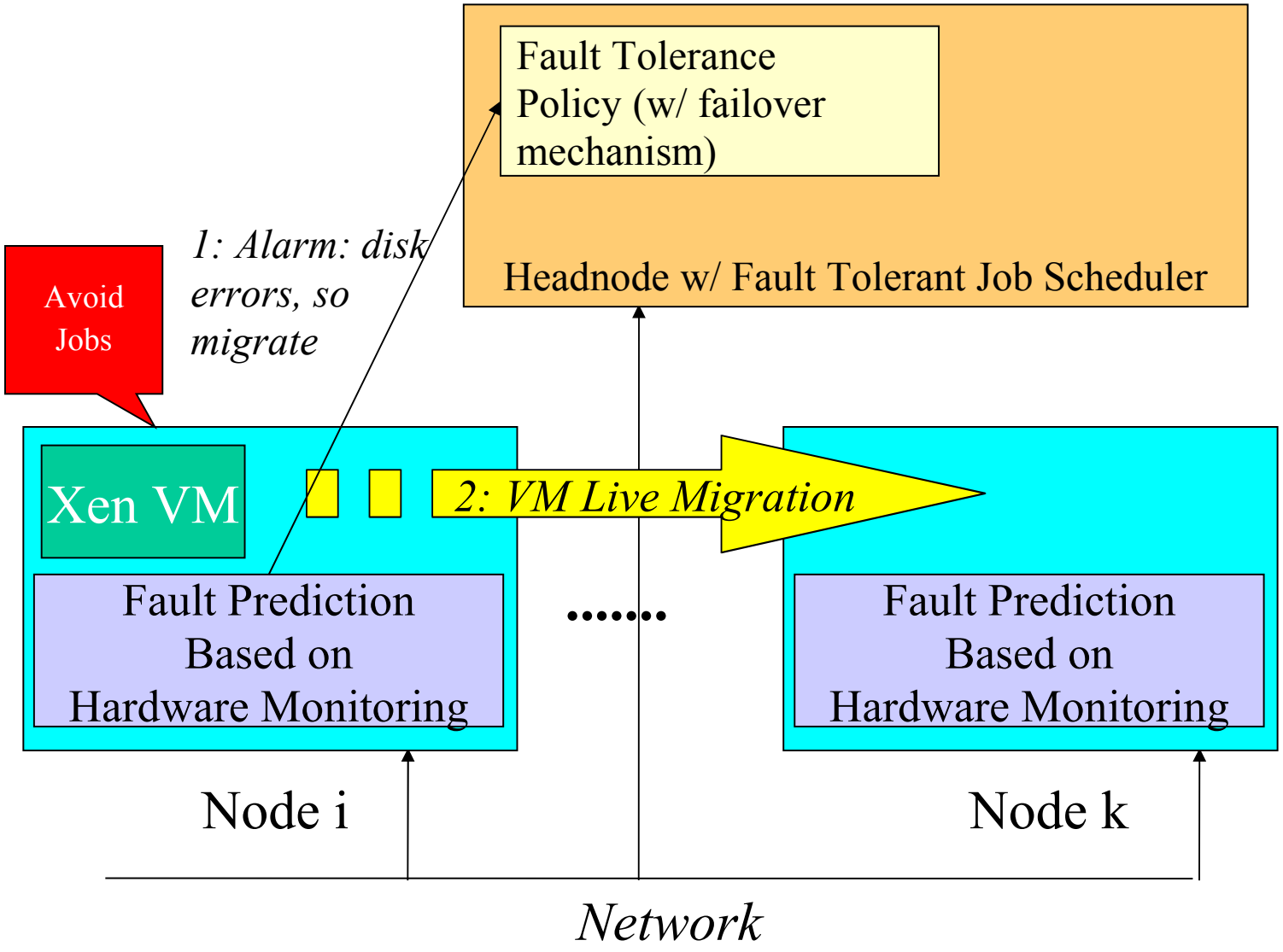
VM Checkpoint / Restart

- Already possible to checkpoint the memory (memory dump in a file)
- File system checkpoint/restart

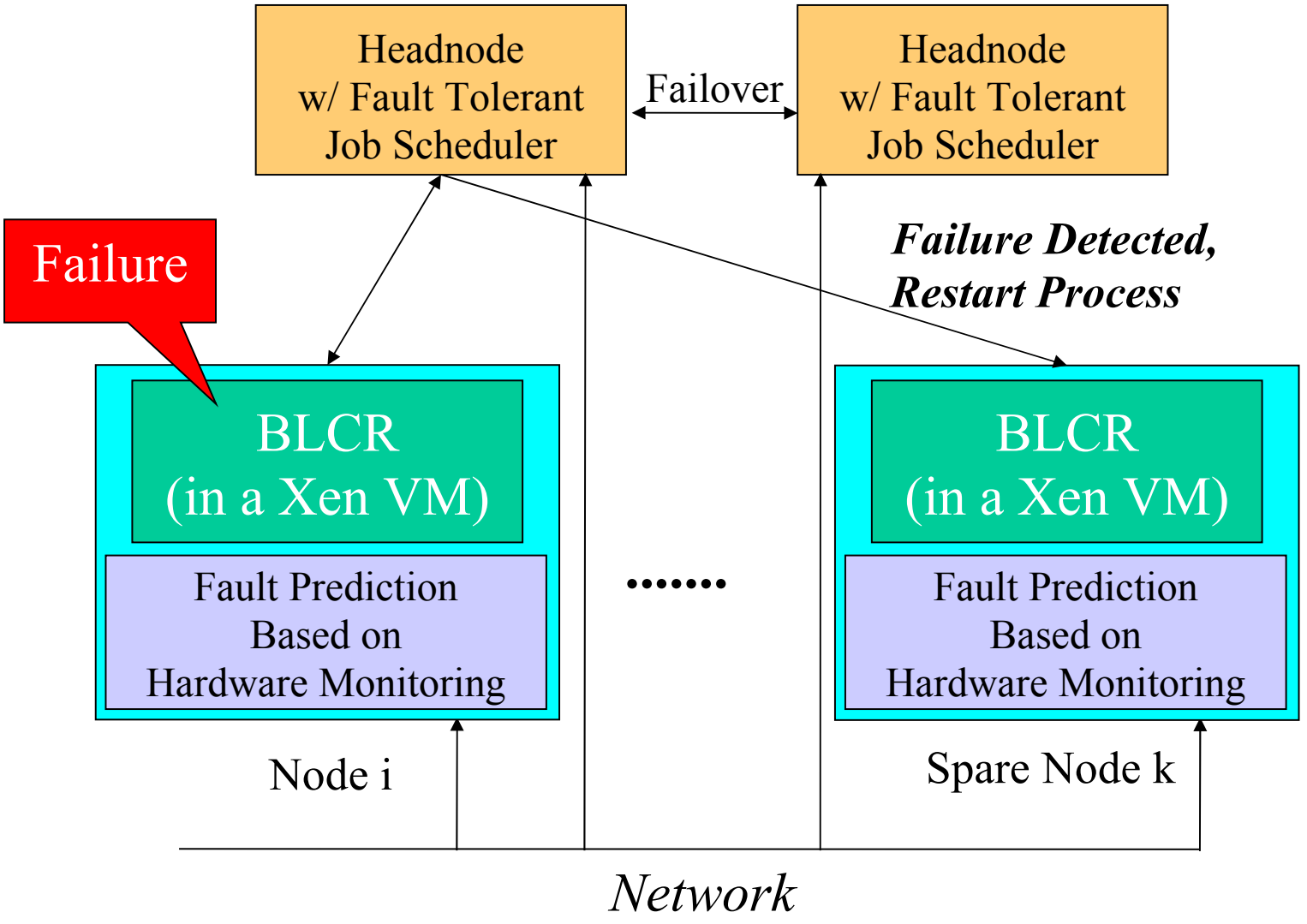


- Collaboration with LATech (Box Leangsuksun)

Pro-active Fault Tolerance



Reactive Fault Tolerance



System Management & Administration

Resource Management

- 2 different main challenges
  - System partitioning (becomes critical with multi-cores)
  - Application deployment
- Virtualization benefits
  - easily enable system partitioning
  - simplify the resource exposure
- Application/VMs deployment
  - on demand
  - before application deployment
- Example: Distributed Virtual Clustering (ASU / Cluster Resources Inc.)
  - Extension of MOAB for the virtualization support

I/O & Storage

I/O & Storage

- Critical for most of the HPC applications (communications, access to storage on service nodes)
- Access to the bare hardware
  - currently through the Host OS (isolation)
  - implies an overhead
- Possible solutions
  - VMM-bypass (direct mapping of resource into VMs)
  - Remote Direct Memory Access (RDMA)

Impact of Virtualization for HPC

- **Programming paradigm**
  - **Challenges: How to move data to/from the application? How to parallelize applications to hundreds or even thousand of nodes? How to checkpoint/restart applications in order to guarantee resiliency?**
  - **Opportunities: implicit communications - move the application to data; change the resource exposure**
- **Application development**
  - **emulation vs. virtualization; OS adaptation; VSE**
  - **application developers can focus on the science**
- **System administration**
  - **separate system administrators and users constraints (VSE)**
- **Foster research and education**
  - **ease research in architecture & operating systems research**

Conclusion

- **Virtualization for HPC implies several challenges**
  - **a hypervisor suitable for HPC (i.e., with a small system footprint)**
  - **the support of virtual system environments**
  - **the support of high availability and fault tolerance capabilities**
  - **the support of advanced resource management capabilities**
  - **the use of system-level virtualization for resource management**
  - **the support of efficient I/O mechanisms and storage solutions for virtualized environment.**
- **No current commercial solution provides such capabilities**
- **Virtualization solution are still immature for HPC (even if studied since the 70's) and still a lot of research to do**

Acknowledgement

Ideas presented in this document are based on discussions with those attending the September 20-21, 2006, Nashville (Tennessee, USA) meeting on the role of virtualization in high performance computing. Attendees included: Stephen L. Scott – meeting chair (Oak Ridge National Laboratory), Barney Maccabe (University of New Mexico), Ron Brightwell (Sandia National Laboratory), Peter A. Dinda (Northwestern University), D.K. Panda (Ohio State University), Christian Engelmann (Oak Ridge National Laboratory), Ada Gavrilovska (Georgia Tech), Geoffroy Vallee (Oak Ridge National Laboratory), Greg Bronevetsky (Lawrence Livermore National Laboratory), Frank Mueller (North Carolina State University), Dan Stanzione (Arizona State University), Hong Ong (Oak Ridge National Laboratory), Seetharami R. Seelam (University of Texas at El Paso), Chokchai (Box) Leangsuksun (Louisiana Tech University), Sudharshan Vazhkudai (Oak Ridge National Laboratory), David Jackson (Cluster Resources Inc.), and Thomas Naughton (Oak Ridge National Laboratory).

We thank them for their time and suggestions for this document.

Contacts

Stephen L. Scott
Computer Science Research Group
Computer Science and Mathematics Division
scottsl@ornl.ornl

Geoffroy Vallee
Computer Science Research Group
Computer Science and Mathematics Division
valleegr@ornl.ornl

*Join us at **ACM HPCVirt'08**, in conjuction with **EuroSys 2008***
March 31, Glasgow, Scotland
http://www.csm.ornl.gov/srt/hpcvirt08/

OAK RIDGE

Questions?