MCEM: Multi-Level Cooperative Exception Model for HPC Workflows

Stephen Herbein, David Domyancic, Paul Minner, Ignacio Laguna,
Rafael Ferreira da Silva , Dong H. Ahn

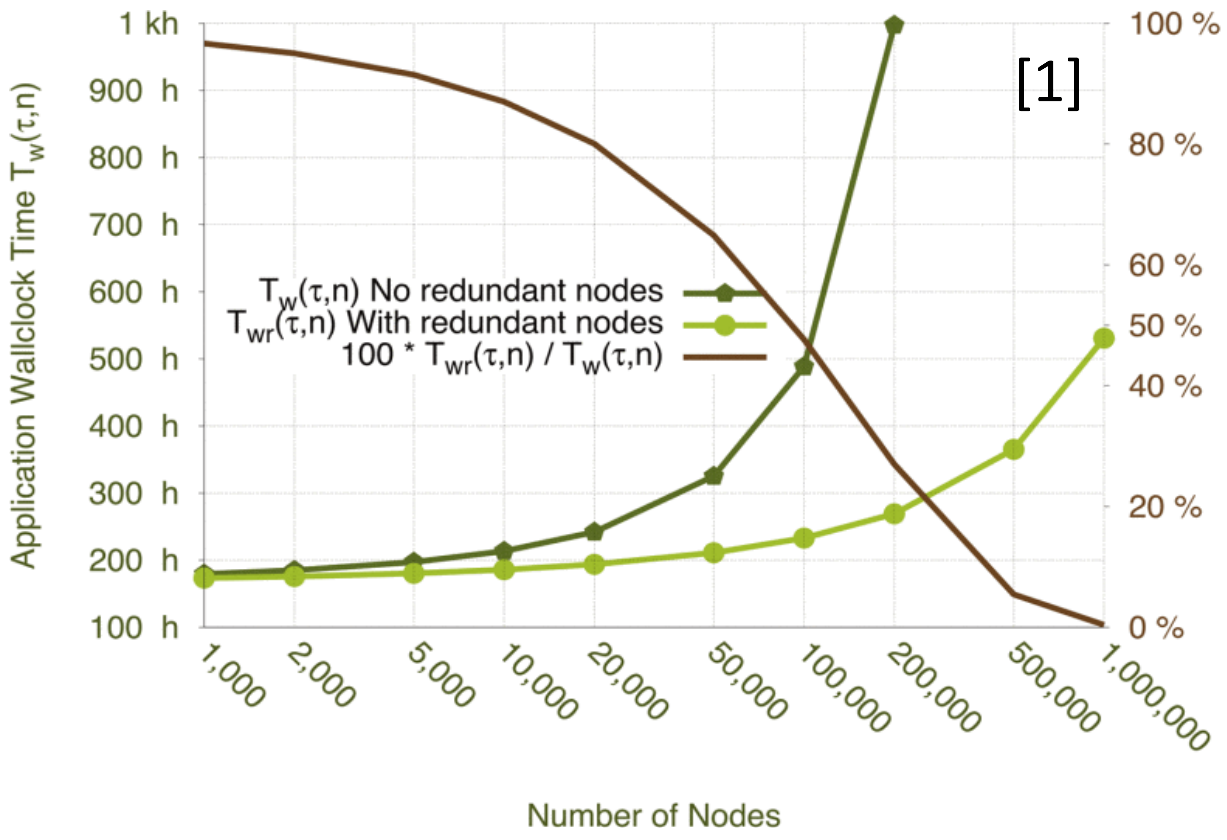
June, 2019

LLNL-PRES-779103

This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under contract DE-AC52-07NA27344. Lawrence Livermore National Security, LLC

Lawrence Livermore National Laboratory

Fault-Tolerance in HPC



- The MTBF of our systems is shrinking
- The cost of checkpoint/restart is becoming prohibitively expensive
- The problem will only get worse with the inclusion of GPUs and node-local SSDs

Fault-Tolerance is becoming increasingly important

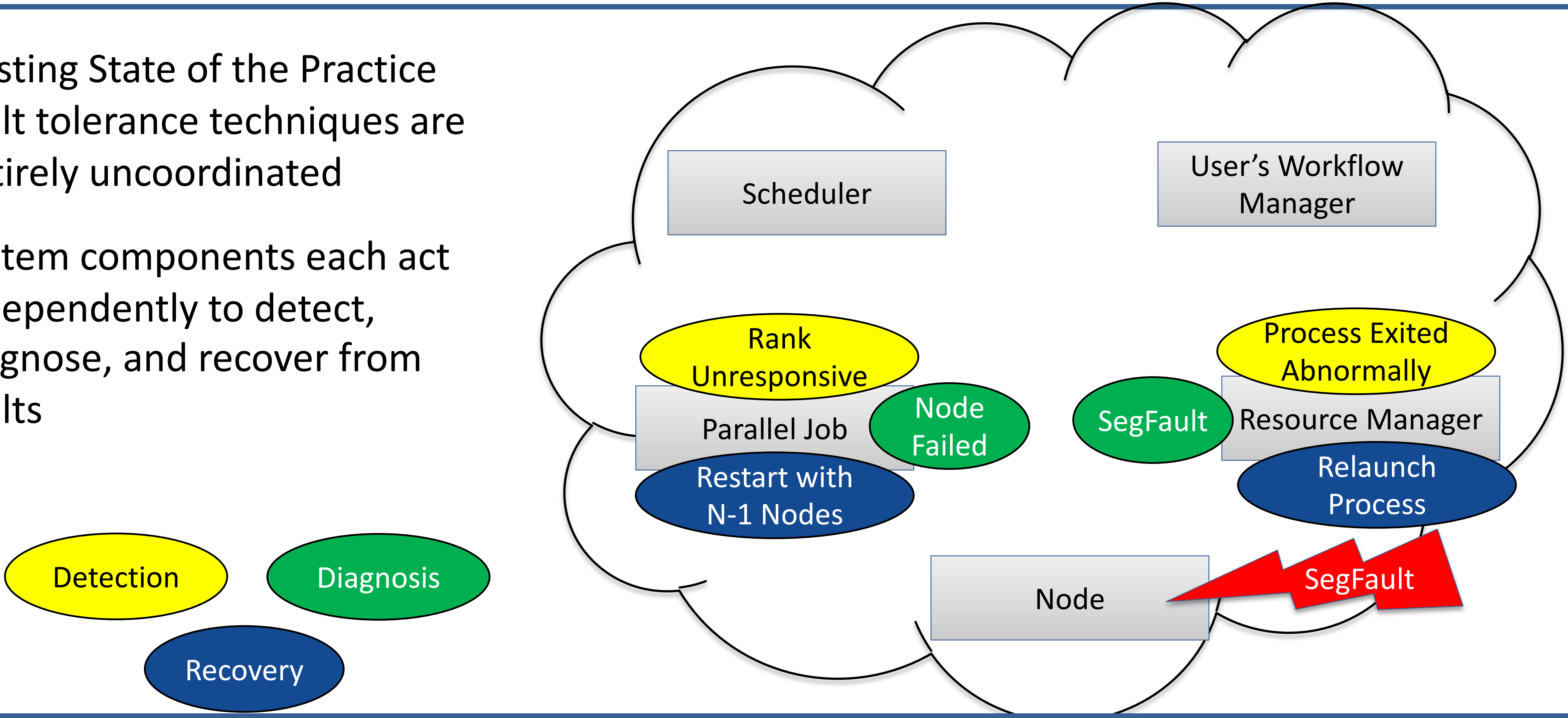
[1] R. Riesen, K. Ferreira and J. Stearley, "See applications run and throughput jump: The case for redundant computing in HPC," *2010 International Conference on Dependable Systems and Networks Workshops (DSN-W)*, Chicago, IL, 2010, pp. 29-34

Fault-Tolerance Primitives

- Detection
  - the observation of a fault, error, or degradation
- Isolation/Diagnosis
  - the identification of the root cause of the detected fault
- Recovery
  - the remediation of the fault by affected components

Fault Tolerance: State of the Practice

- Existing State of the Practice fault tolerance techniques are entirely uncoordinated
- System components each act independently to detect, diagnose, and recover from faults



Lack of coordination results in undetected faults and inefficiency

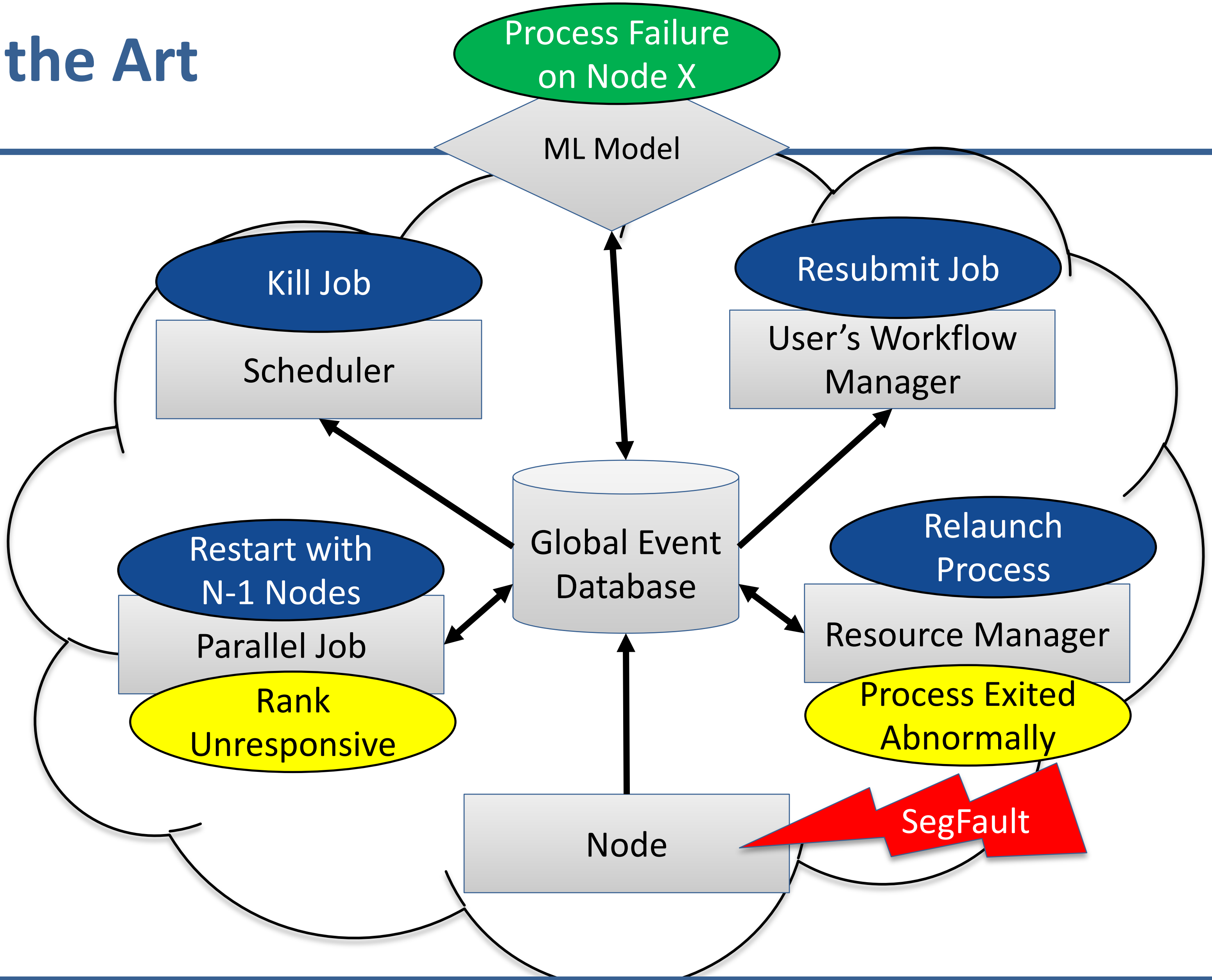
Fault Tolerance: State of the Art

- Components coordinate to detect and diagnose faults
- System components each perform their own uncoordinated recovery actions
- These actions are usually redundant and sometimes contradictory

Detection

Diagnosis

Recovery



Process Failure on Node X

ML Model

Kill Job

Scheduler

Resubmit Job

User’s Workflow Manager

Global Event Database

Restart with N-1 Nodes

Parallel Job

Rank Unresponsive

Relaunch Process

Resource Manager

Process Exited Abnormally

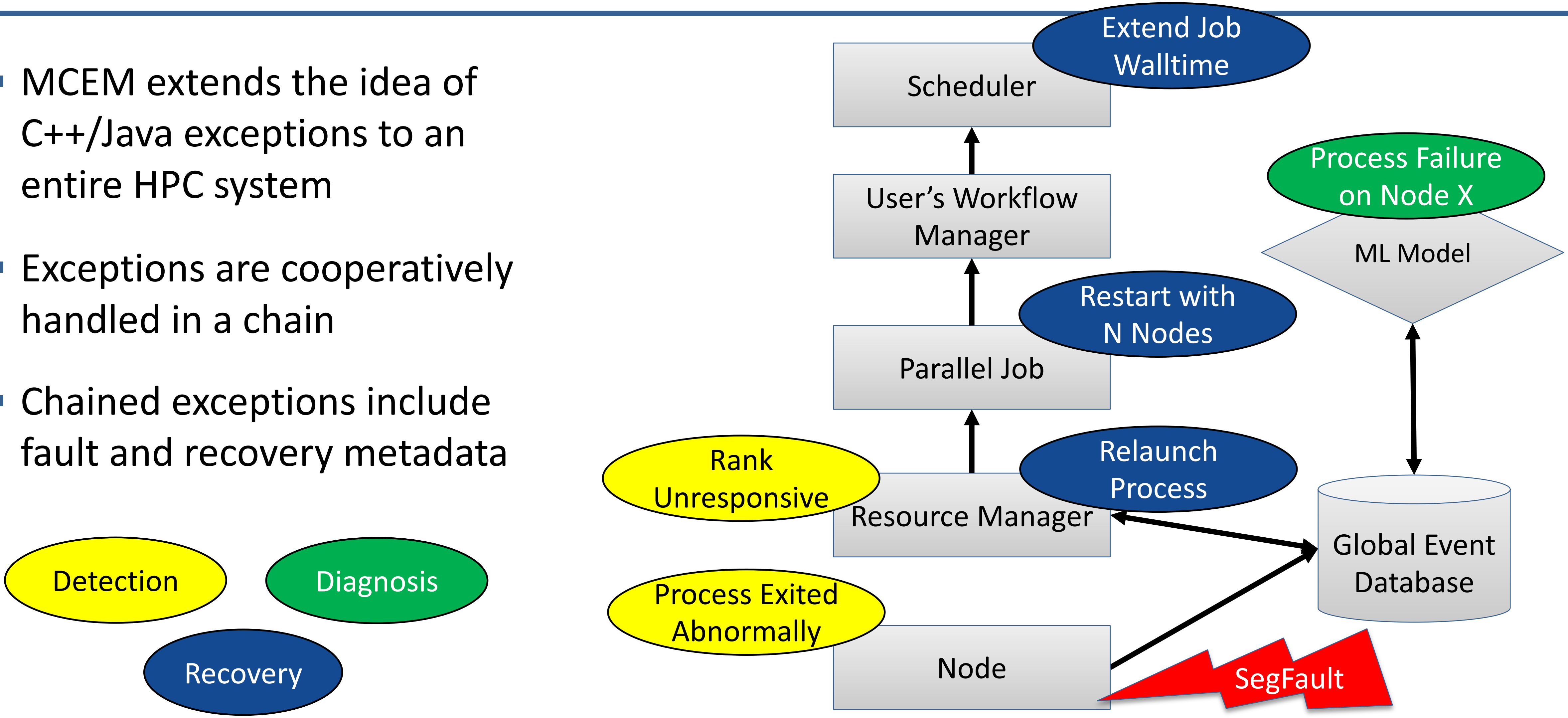
Node

SegFault

Lack of coordinated recovery results in suboptimal and redundant work

MCEM: Multi-Level Cooperative Exception Model

- MCEM extends the idea of C++/Java exceptions to an entire HPC system
- Exceptions are cooperatively handled in a chain
- Chained exceptions include fault and recovery metadata



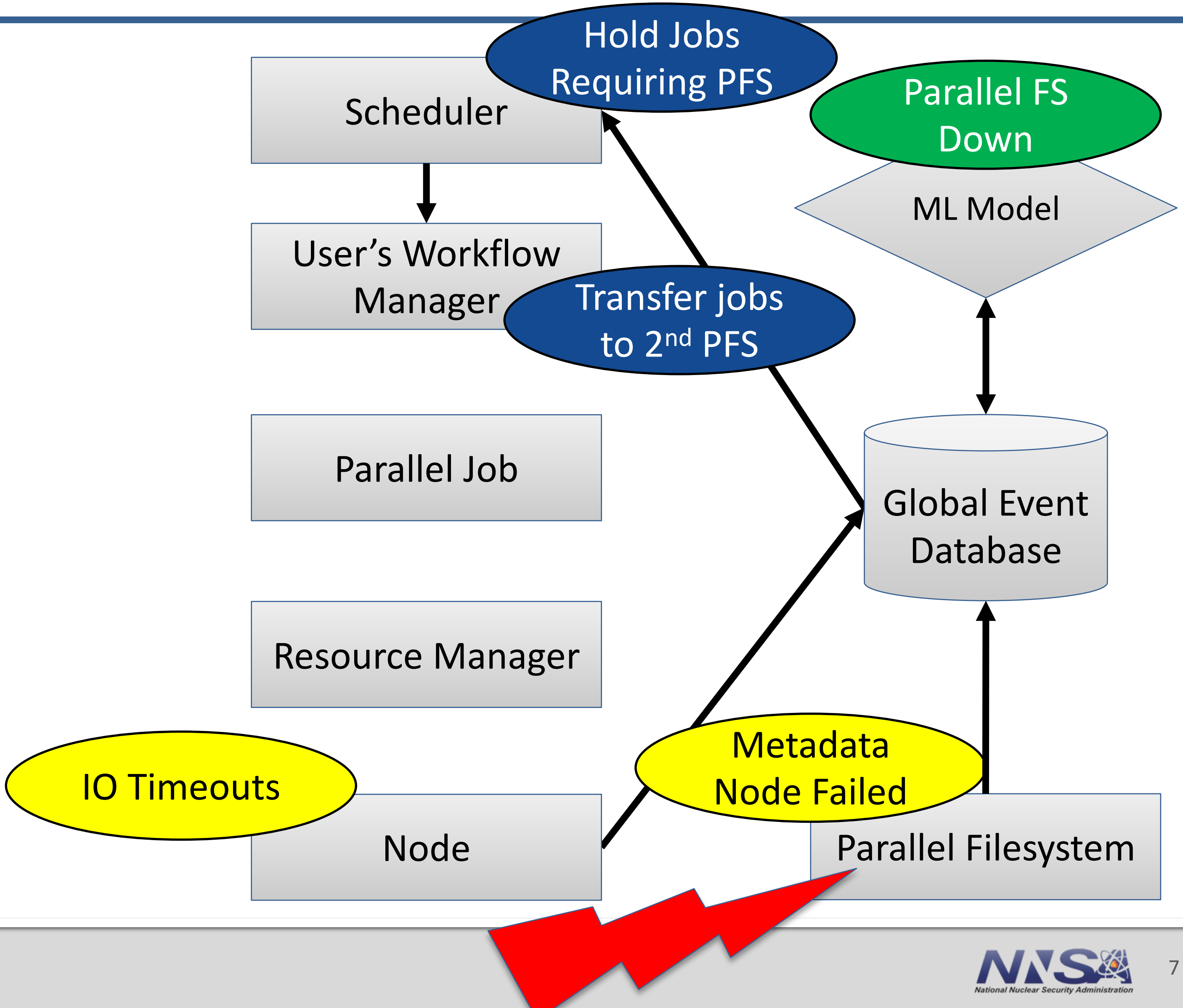
MCEM: Global Exceptions

- Propagating up works well for exceptions originating from a single, isolated resource (i.e., *local exception*)
- Reverse propagation direction for exceptions originating from a shared resource (i.e., *global exception*)

Detection

Diagnosis

Recovery



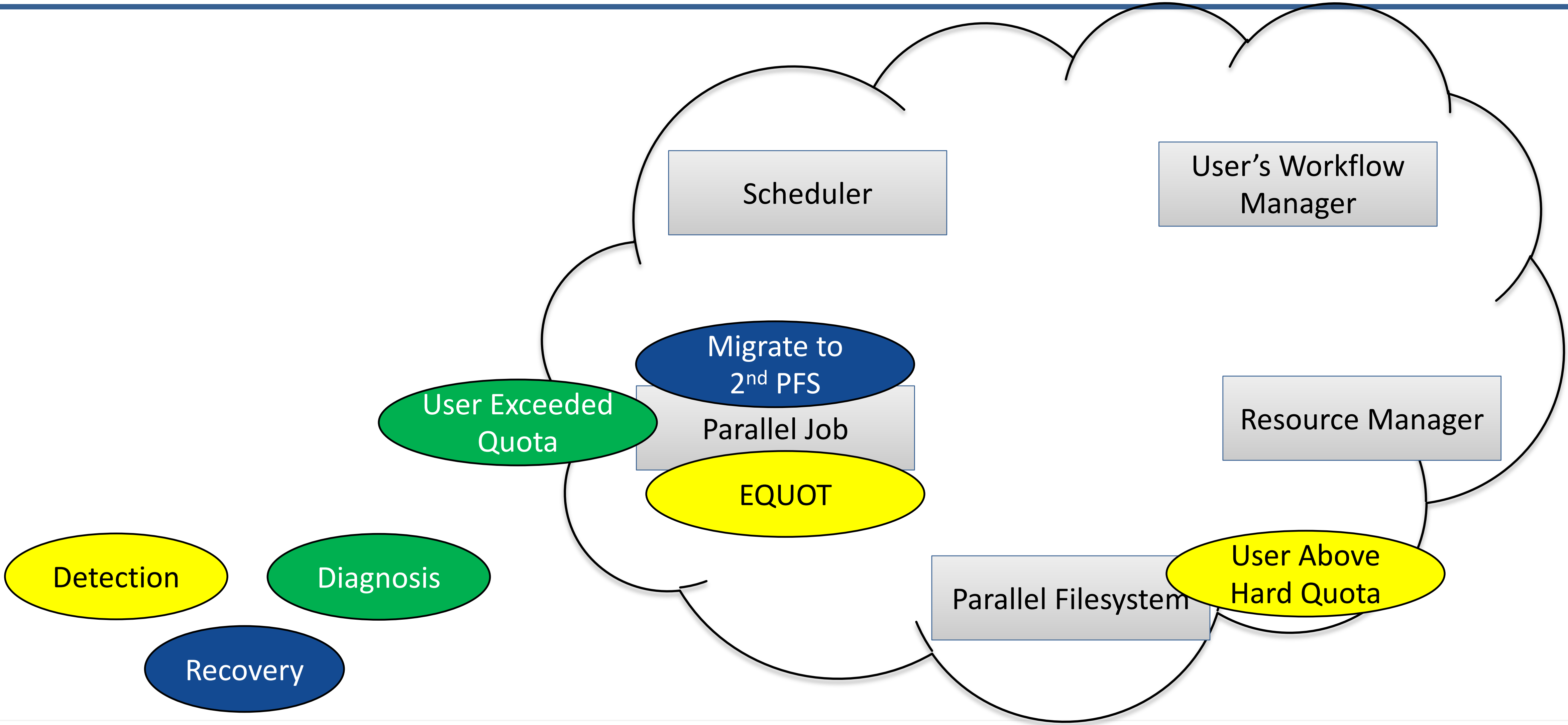
MCEM: Fault Model

- Hard faults
  - Segmentation Faults, Node Failures, Network Link Failure, PFS Down, User Exceeded Disk Quota
- Soft faults
  - Network or PFS performance degraded, User Approaching Disk Quota
- Fault length
  - Effects must last long enough to be reliably detected, isolated, and recovered from – O(minutes)

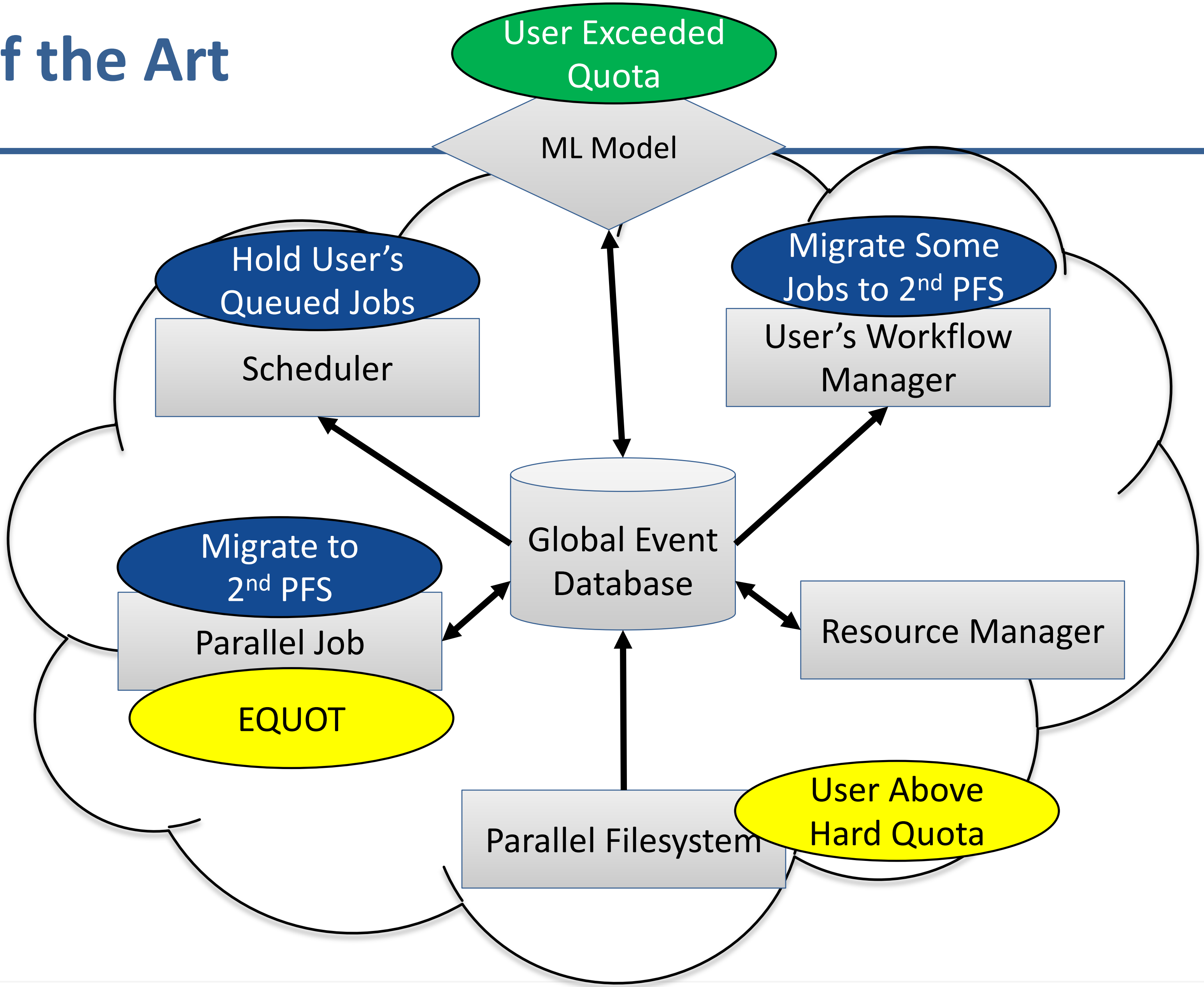
MCEM Exception Recovery Examples

Failure Type	Resource Manager	Parallel Job	Workflow Manager	Scheduler
Parallel Launcher Failure	--	--	Retry job (transient) Log system error (permanent)	--
Application Failure (i.e., mesh tangling)	--	--	Launch mesh relaxation job	--
Process Failure	Relaunch Process	Restart w/ N ranks	--	Grant job addt'l time
Node Failure	Mark node down	Restart w/ N-1 ranks OR req addt'l node	--	Grant job addt'l node
User Approaching or Exceeding Disk Quota	--	--	Migrate some/all workflow jobs to secondary filesystem	Hold queued jobs requiring PFS access

Quota Exceeded: State of the Practice



Quota Exceeded: State of the Art

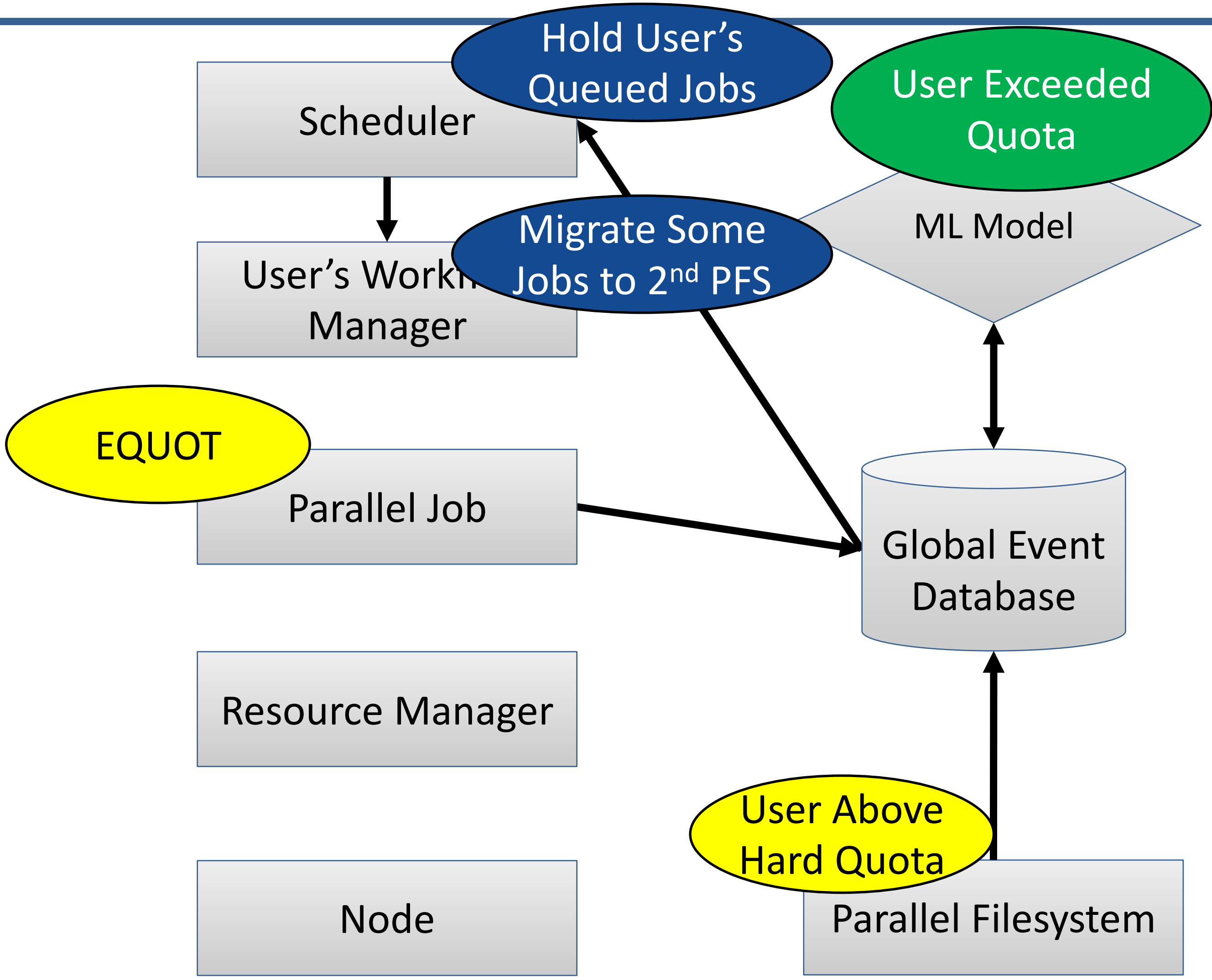


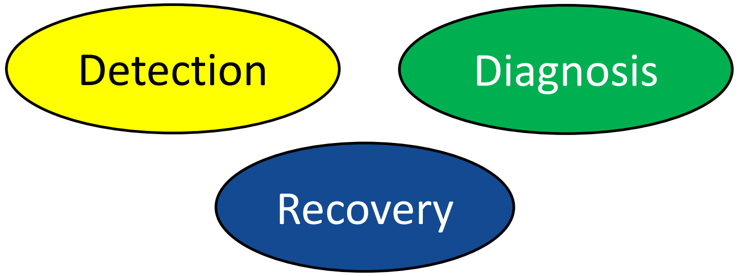
Detection

Diagnosis

Recovery

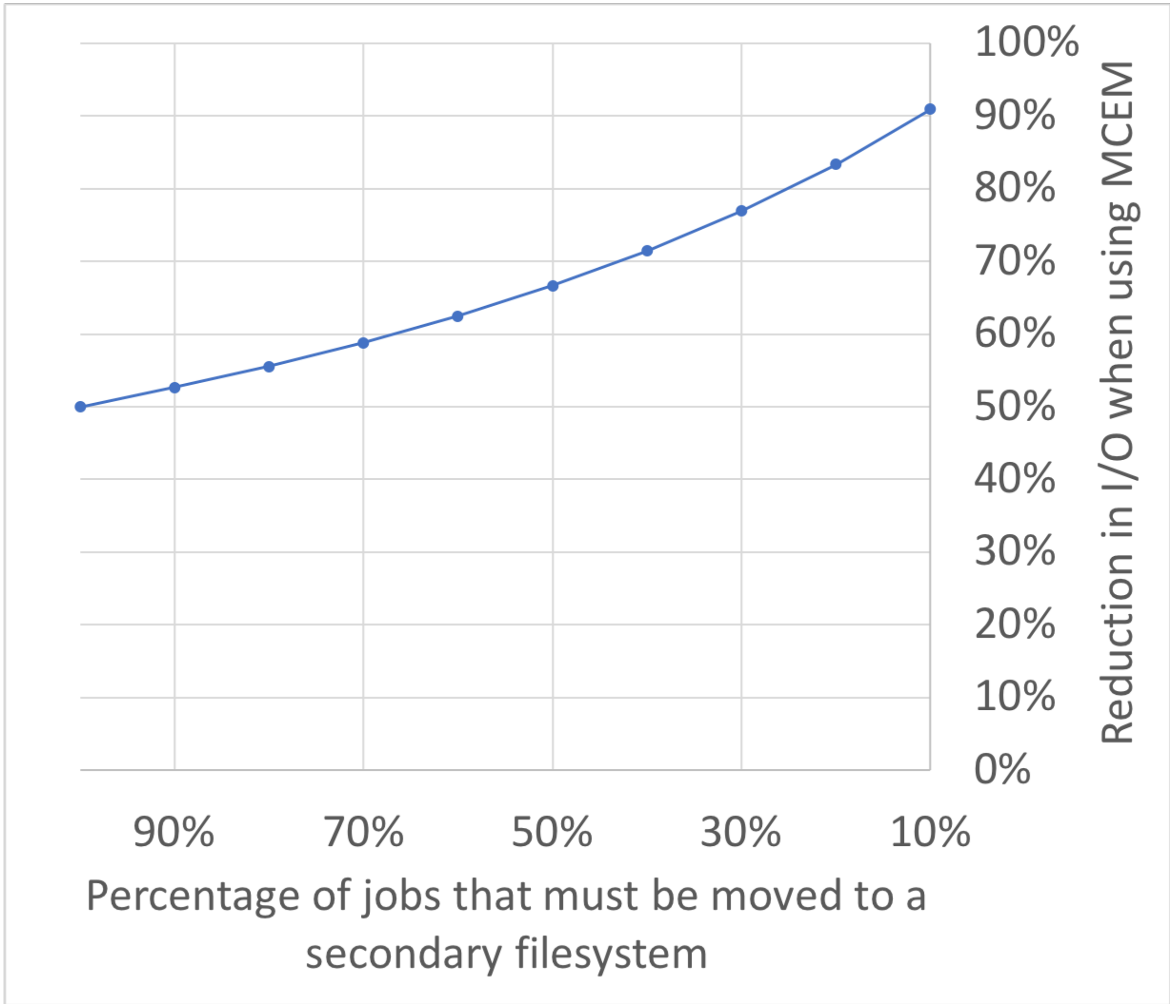
Quota Exceeded: MCEM





Evaluation

- In SOA, parallel applications all transition to 2nd filesystem, and the WFM re-transitions some/all of the jobs
- MCEM allows the WFM to only move the minimal subset of jobs exactly once



MCEM can reduce IO by up to 90%

Implementation: Resource Manager

- Why to implement within the system RM
  - Communication already implemented and fault-tolerant (hopefully)
  - Can be a plugin/module, result in less code to write and audit

- Why not to implement within the system RM
  - If the RM daemon dies, so does MCEM
  - RM failures then become potentially undetectable and certainly unrecoverable

Implementation: Runtime Interface

- Flux
  - flux job raise –severity=1 –type=“segmentation fault” $ID ’{“rank”: “262”, “pid”: 1182, “node”: ”quartz454”}’
  - flux job eventlog $ID
  - flux_event_subscribe (h, "job-exception")

- PMIx
  - PMIx_Notify_event
  - PMIx_Register_event_handler
    - Supports registering a handler for multiple events, simultaneously
    - “Multi-code” handlers always execute after “single-code” handlers
    - Supports specifying relative handler precedence within a “category”

Acknowledgements

- Co-Authors
  - — David Domyancic
  - — Paul Minner
  - — Ignacio Laguna
  - — Rafael Ferreira da Silva
  - — Dong H. Ahn

- Flux Team
  - — Ned Bass
  - — Al Chu
  - — Jim Garlick
  - — Mark Grondona
  - — Tapasya Patki
  - — Tom Scogland
  - — Becky Springmeyer

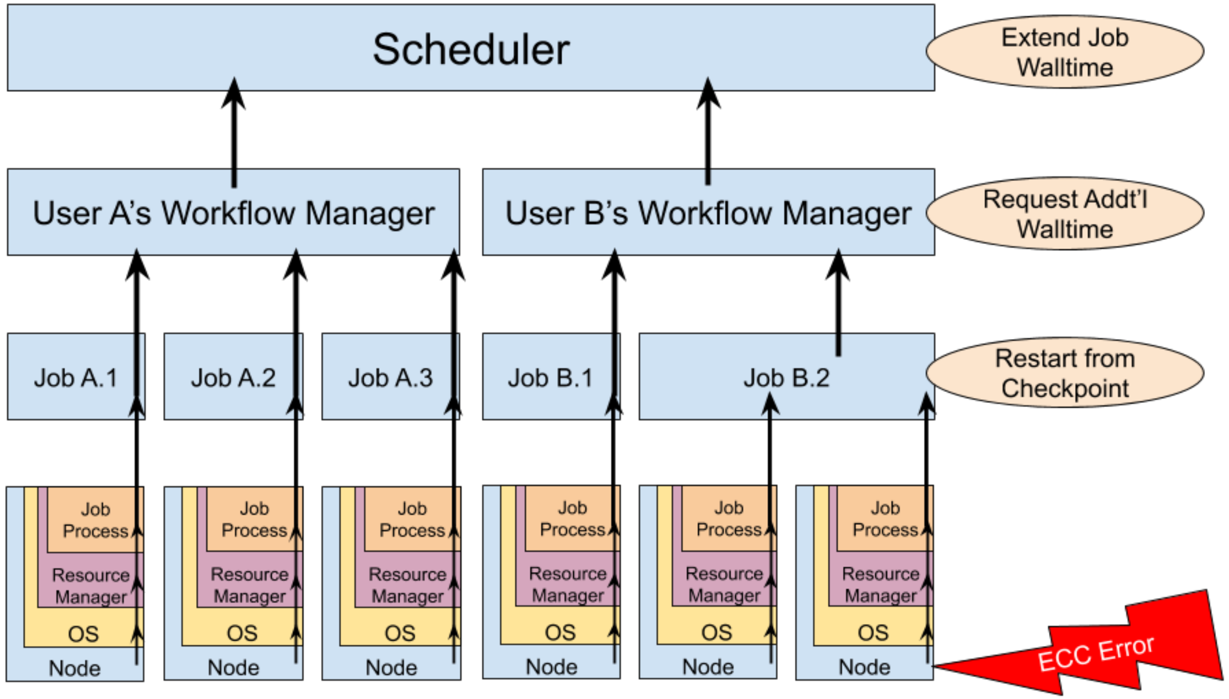
Lawrence Livermore
National Laboratory

Disclaimer
This document was prepared as an account of work sponsored by an agency of the United States government. Neither the United States government nor Lawrence Livermore National Security, LLC, nor any of their employees makes any warranty, expressed or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States government or Lawrence Livermore National Security, LLC. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States government or Lawrence Livermore National Security, LLC, and shall not be used for advertising or product endorsement purposes.

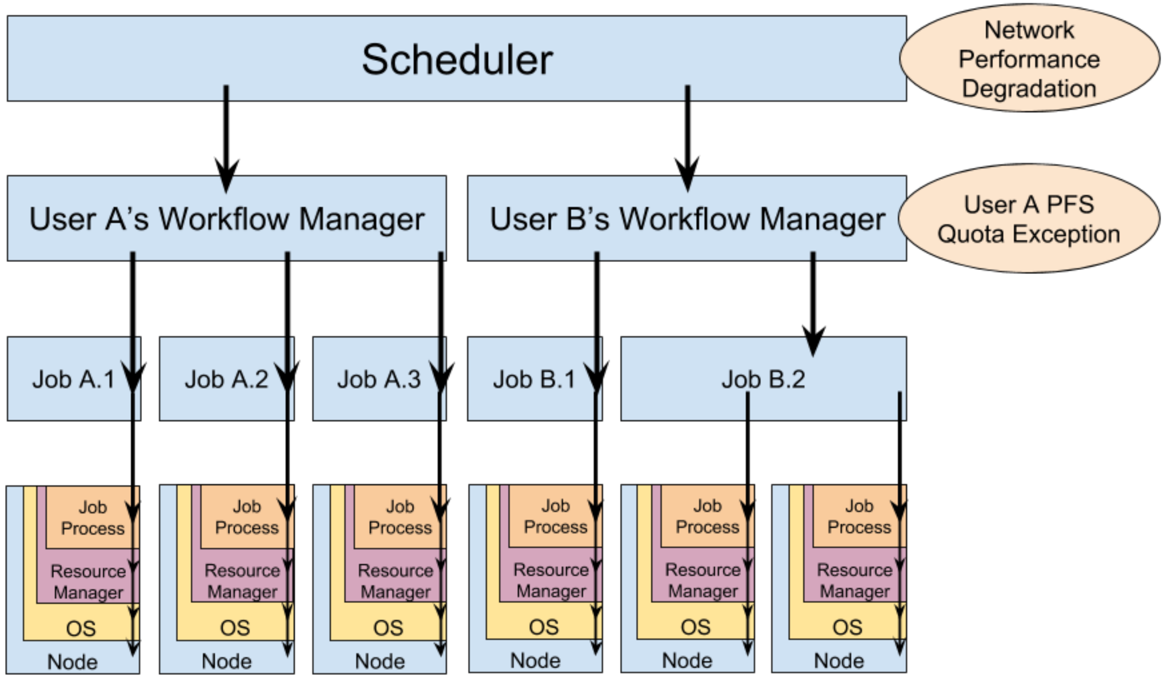
Backup Slides

MCEM’s Exception Propagation Order

Local Exceptions



Global Exceptions


Scheduler
Network Performance Degradation
User A’s Workflow Manager
User B’s Workflow Manager
User A PFS Quota Exception
Job A.1
Job A.2
Job A.3
Job B.1
Job B.2
Job Process
Resource Manager
OS
Node