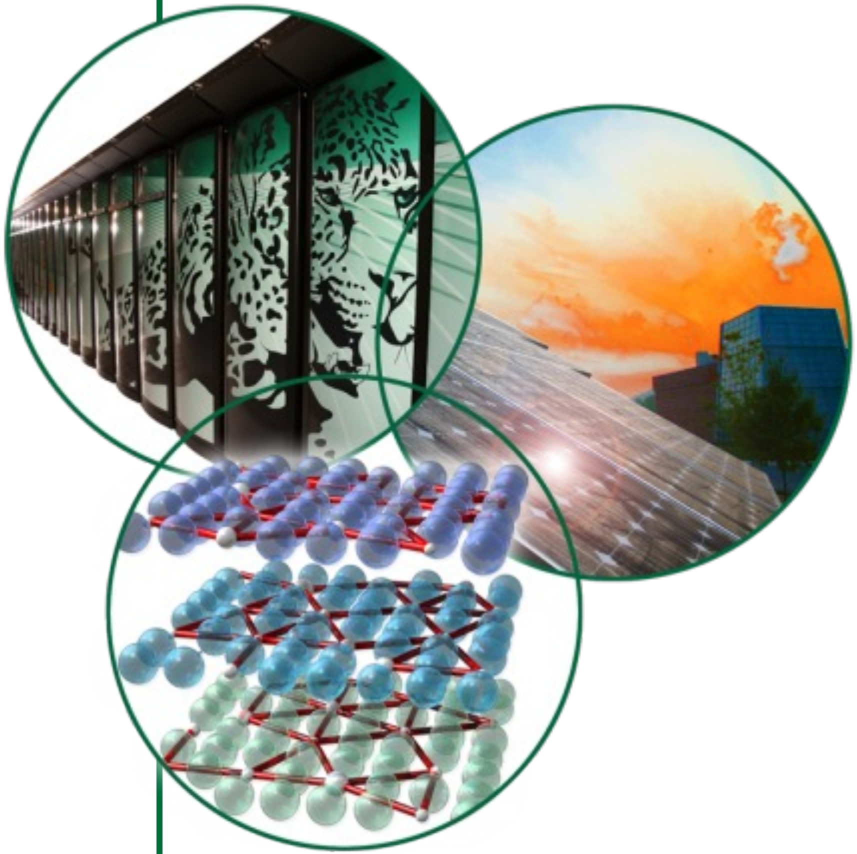
Toward A Fault Model And Resilience Design Patterns For Extreme Scale Systems

Christian Engelmann

System Software Team Lead
Computer Science Research Group
Computer Science and Mathematics Division
Oak Ridge National Laboratory

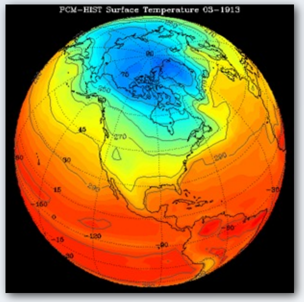


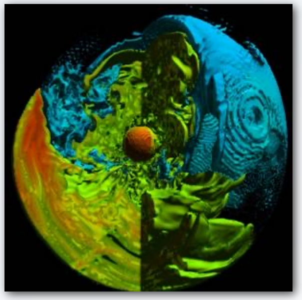
Resilience Workshop @ Euro-Par 2015, Vienna, Austria, August 24, 2015.

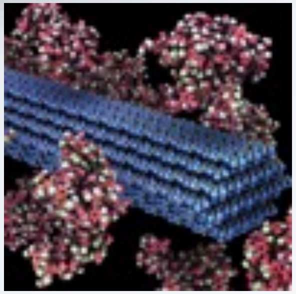
U.S. DEPARTMENT OF ENERGY

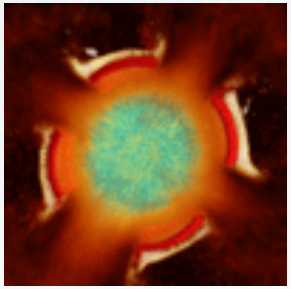
OAK RIDGE NATIONAL LABORATORY
MANAGED BY UT-BATTELLE FOR THE DEPARTMENT OF ENERGY

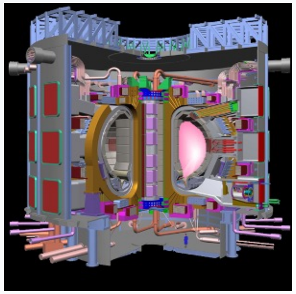
Scientific Computing and Simulation at ORNL

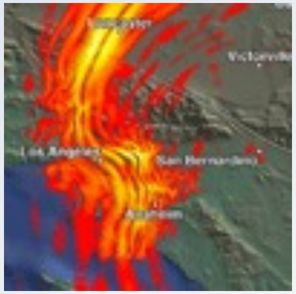


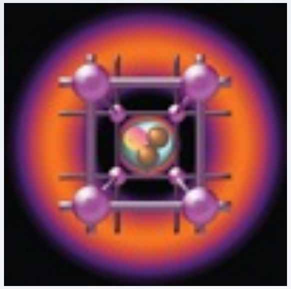


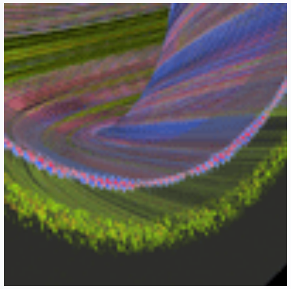








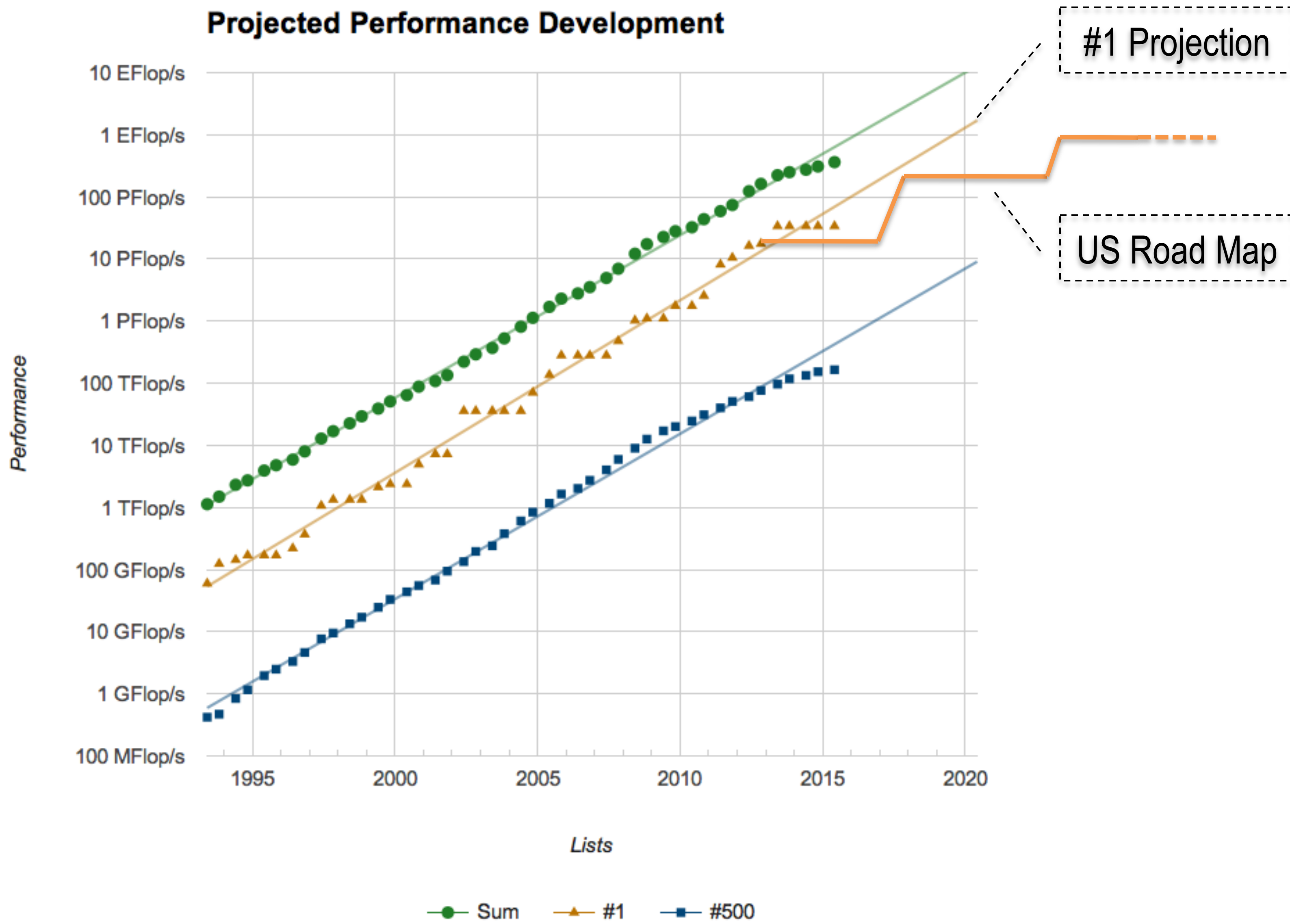




Motivation

- At the forefront of extreme-scale scientific computing
  - Titan at ORNL: Currently 2nd fastest supercomputer in the world
  - 560,640 cores (AMD Opteron + NVIDIA Kepler GPUs, 17.6 PFlops)
- We are on road to exascale computing: 1,000 Pflop/s by 2023
  - Potentially ***billions*** of cores at exascale
- There are several major challenges:
  - ***Power consumption***: Envelope of ~20-40 MW (drives everything else)
  - ***Programmability***: Accelerators and PIM-like architectures
  - ***Performance***: Extreme-scale parallelism (up to 1B hardware threads)
  - ***Data movement***: Complex memory hierarchy and locality
  - ***Data management***: Too much data to track and store
  - ***Resilience***: Faults will occur continuously

Top 500 List of Supercomputers



C. Engelmann. Toward A Fault Model And Resilience Design Patterns For Extreme Scale Systems. Resilience Workshop 2015.

Why is Resilience a Challenge?

- Smaller and smaller process technology
  - Unknown aging effects
  - Increased variability
  - Increased soft error vulnerability
- Near-threshold voltage to achieve energy efficiency
  - Increased soft error vulnerability
  - Decreased noise immunity
- System MTTI decreases as component count increases
  - Can not offset component growth with reliability improvements anymore
- System software complexity
  - Most failures are due to system software (e.g. parallel file system)

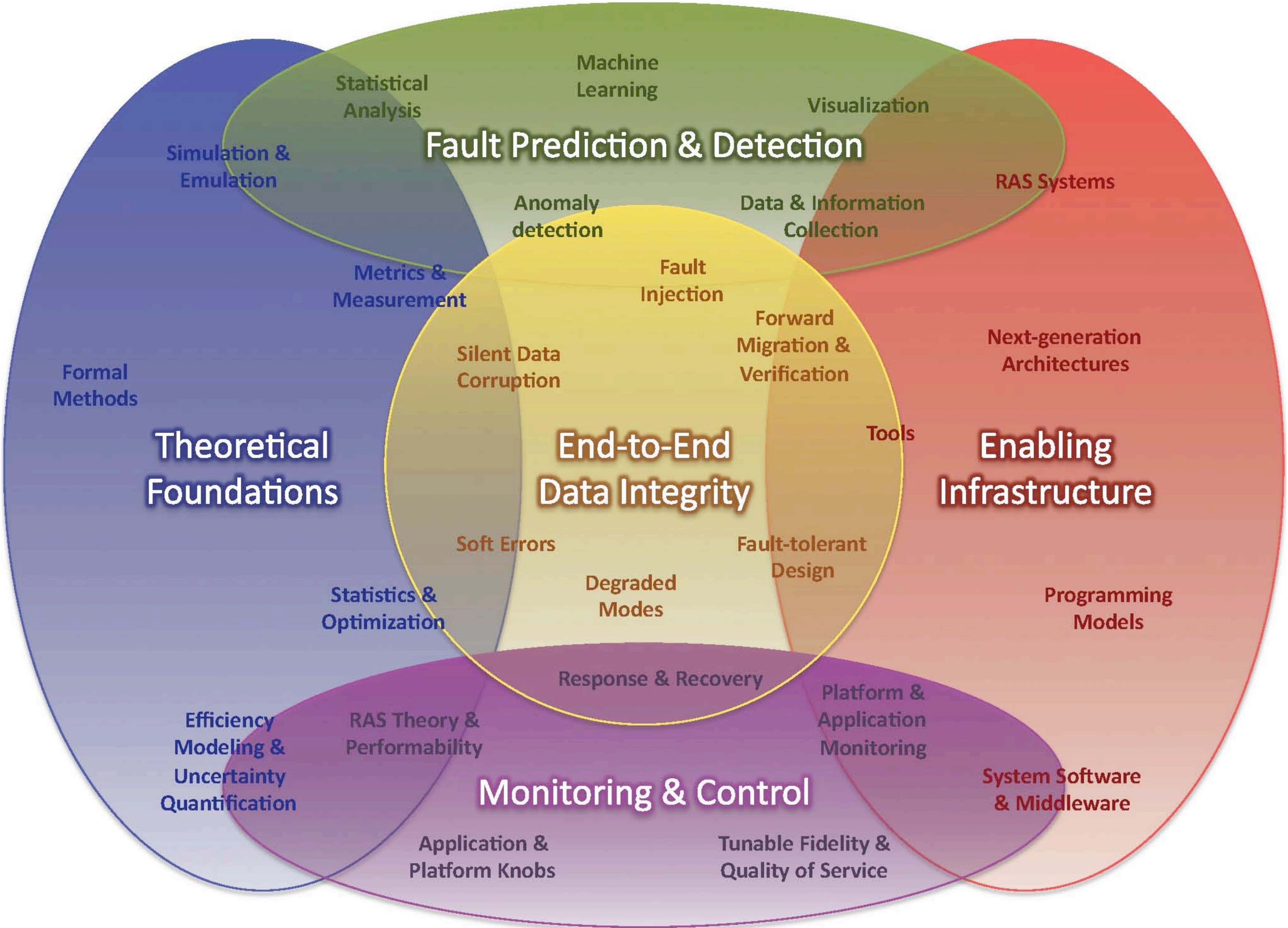
C. Engelmann. Toward A Fault Model And Resilience Design Patterns For Extreme Scale Systems. Resilience Workshop 2015.

Why is Resilience a Challenge?

- NO: It's a cost problem
  - This isn't NASA's Moon Program ($25.4 billion in 1973 Dollars - $170 billion in 2005 Dollars - over an 11-year period)

- The challenge is to build a reliable system within a given cost budget that achieves the expected performance

Key Areas of HPC Resilience Research



The Monster in the Closet

- Resilience is a critical challenge for extreme-scale HPC
- Solutions have been and are being developed
- **There are still many open questions:**
  - There is no HPC fault model
  - Performance/resilience/power trade-off is not understood holistically
  - Coordination across the stack is missing
  - End-to-end management of resilience is needed
  - Resilient parallel programming models do not exist
  - Operational resilience policies do not go beyond checkpoint/restart

C. Engelmann. Toward A Fault Model And Resilience Design Patterns For Extreme Scale Systems. Resilience Workshop 2015.

Characterizing Faults, Errors, and Failures in Extreme-scale Systems

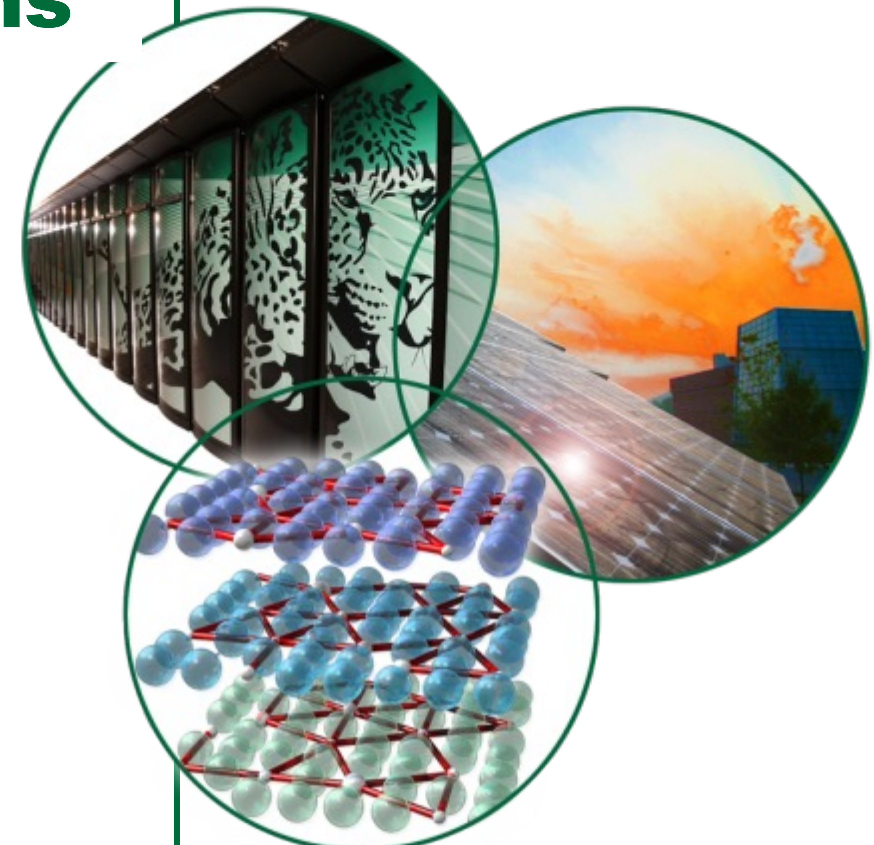
Christian Engelmann
Oak Ridge National Laboratory

Martin Schulz
Lawrence Livermore National Laboratory

Marc Snir
Argonne National Laboratory

Resilience for Extreme Scale Supercomputing Systems Program
Office of Advanced Scientific Research
Office of Science, US Department of Energy

U.S. DEPARTMENT OF ENERGY



Lawrence Livermore National Laboratory
Argonne NATIONAL LABORATORY
OAK RIDGE NATIONAL LABORATORY
MANAGED BY UT-BATTELLE FOR THE DEPARTMENT OF ENERGY

Objectives

- This project identifies, categorizes and models the fault, error and failure properties of DOE systems
- It develops a fault taxonomy, catalog and models that capture the observed and inferred conditions in current systems and extrapolate this knowledge to exascale systems
- The results of this project will provide a clear picture of the fault characteristics in the DOE computing environments and improve resilience through reliable fault detection at an early stage and actionable information for efficient fault mitigation

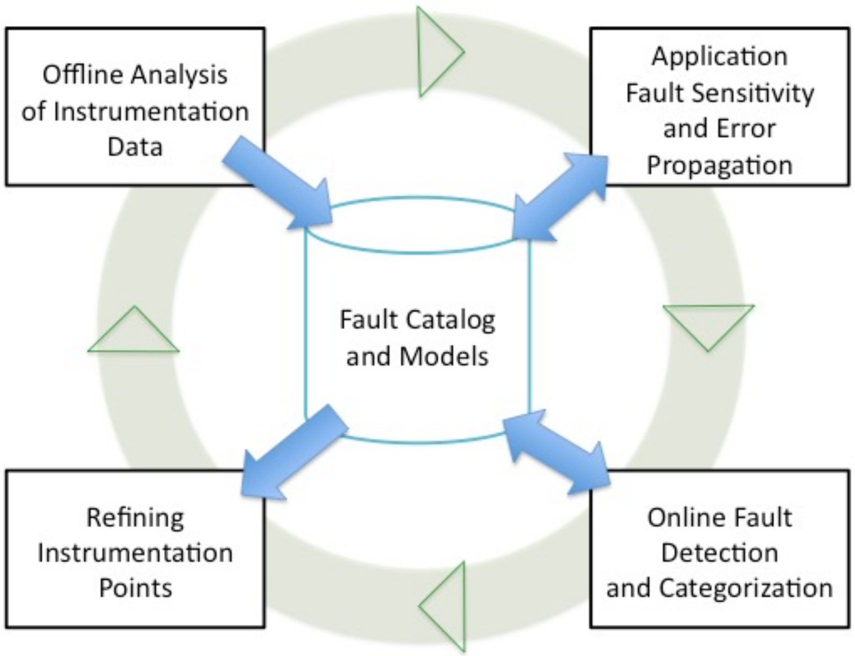
Motivation

- Today's extreme-scale systems succeed in delivering science
- *However, we still lack in understanding the depth and magnitude of the resilience problem*
- Vendors and the HPC community have developed a number of resilience technologies
- *However, proper resilience requires knowing fault root causes, detection delays, error propagation paths, and failure modes*
- Today's systems are heavily instrumented for health monitoring
- *However, fault characterization tools do not use the full spectrum of data, lack in advanced data analytics, do not follow a common taxonomy, and do not consider application health*

C. Engelmann. Toward A Fault Model And Resilience Design Patterns For Extreme Scale Systems. Resilience Workshop 2015.

Approach (1/2)

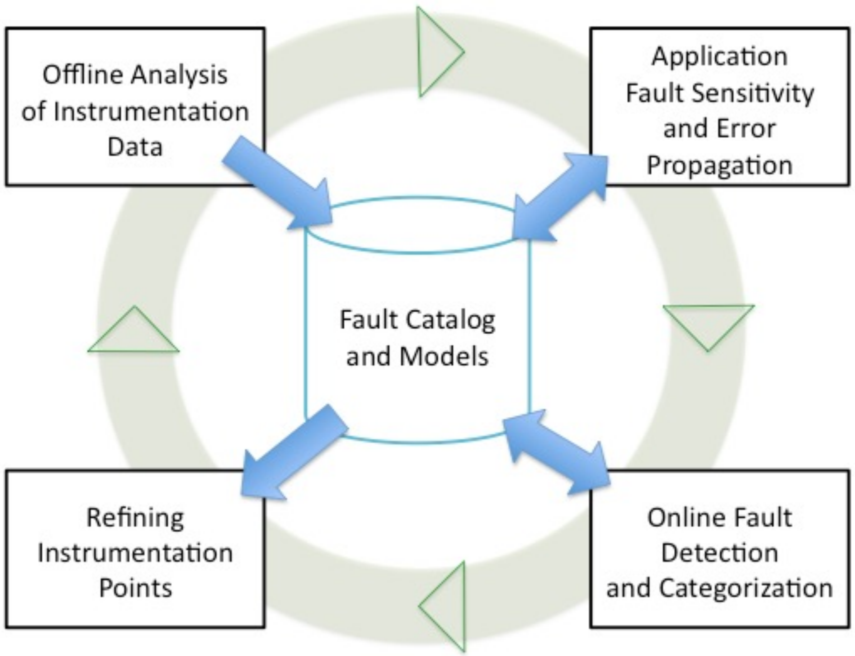
- Create a catalog with a common taxonomy of faults, errors and failures in extreme-scale HPC systems
- Fuse data for offline fault, error and failure identification, categorization, root cause analyses, modeling, and visualization
- Model the impact of faults on, error propagation within, and failure modes of applications in time and space



C. Engelmann. Toward A Fault Model And Resilience Design Patterns For Extreme Scale Systems. Resilience Workshop 2015.

Approach (2/2)

- Develop an online detection and categorization framework that improves the offline approach and visualizes data for feedback
- Identify additional instrumentation points for early detection and more accurate categorization



C. Engelmann. Toward A Fault Model And Resilience Design Patterns For Extreme Scale Systems. Resilience Workshop 2015.

Leveraged Previous Accomplishments

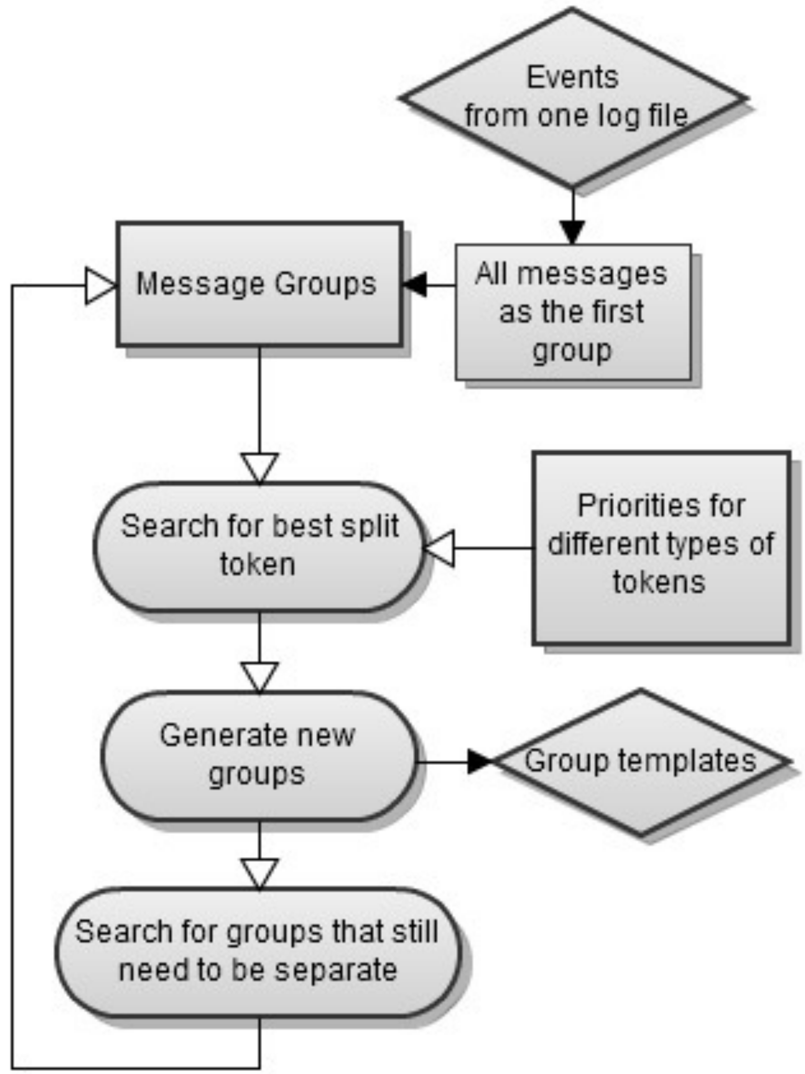
- RAS Data Analysis Through Visually Enhanced Navigation (RAVEN) framework developed at ORNL
  - Offline data analysis and vizualization
- Hierarchical Event Log Organizer (HELO) developed at ANL
  - Offline data analysis
- Event Log Signal Analyzer (ELSA) developed at ANL
  - Fault detection/prediction using signal analysis
- FlipIt LLVM-based fault injection framework developed at ANL
  - Data corruption injection
- GREMLINs framework developed at LLNL
  - Emulation of fault behavior using instrumentation

RAS Data Analysis Through Visually Enhanced Navigation (RAVEN)

C. Engelmann. Toward A Fault Model And Resilience Design Patterns For Extreme Scale Systems. Resilience Workshop 2015.

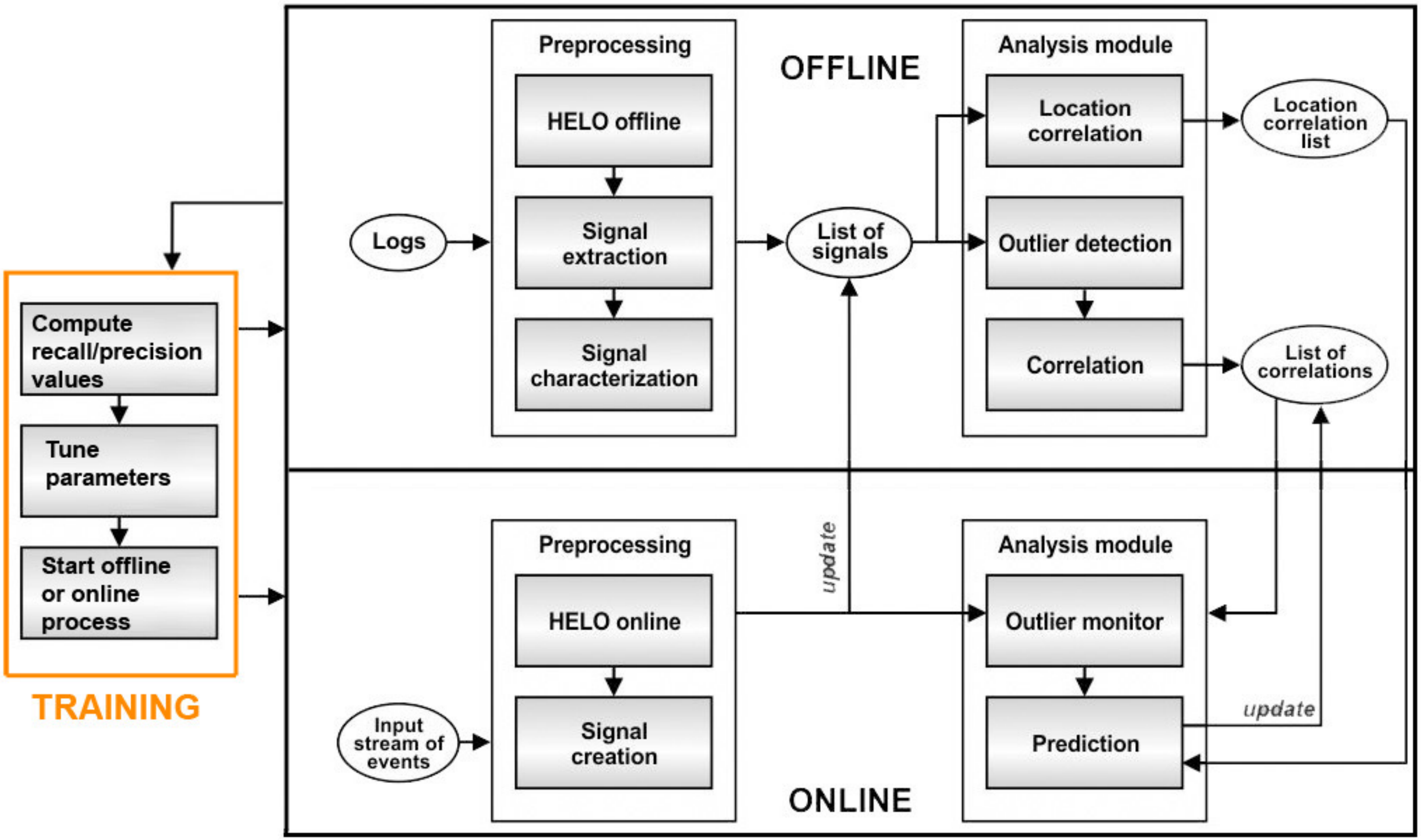
Oak Ridge National Laboratory

Hierarchical Event Log Organizer (HELO)



C. Engelmann. Toward A Fault Model And Resilience Design Patterns For Extreme Scale Systems. Resilience Workshop 2015.

Event Log Signal Analyzer (ELSA)



C. Engelmann. Toward A Fault Model And Resilience Design Patterns For Extreme Scale Systems. Resilience Workshop 2015.

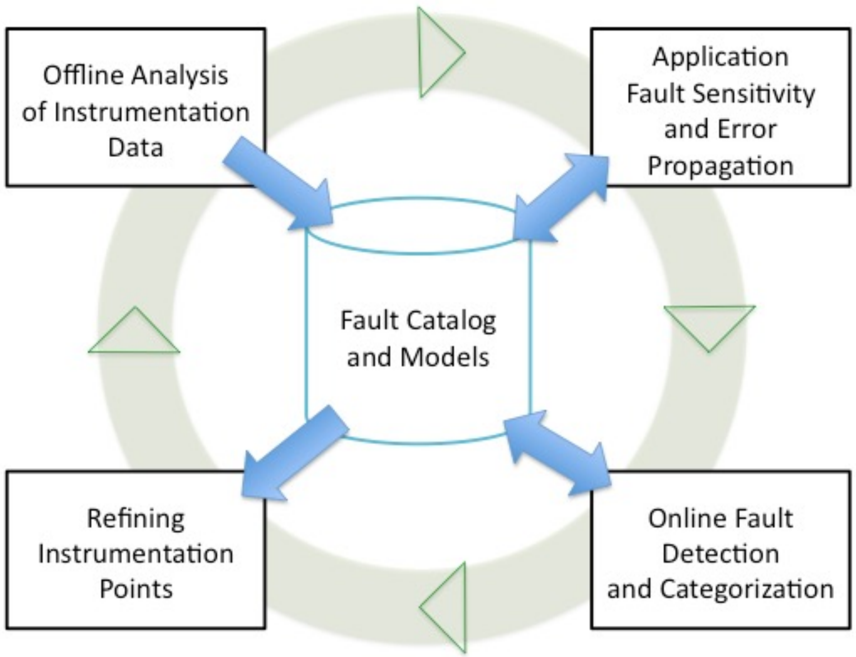
Deliverables

- Year 1
  - Initial fault catalog & models
  - Comprehensive framework with improved offline analysis techniques
  - Infrastructure for realistic fault injection experiments
- Year 2
  - Updated fault catalog & models
  - Characterization of application sensitivity using fault injection
  - Refinement of instrumentation data sets and points for offline analysis
- Year 3
  - Final fault catalog & models
  - Application and system fault and error propagation models
  - Comprehensive online analysis framework with realtime visualization

Team

- Oak Ridge National Laboratory
  - Christian Engelmann (PI)
  - Byung-Hoon (Hoony) Park
  - Devesh Tiwari
- Lawrence Livermore National Laboratory
  - Martin Schulz (Institutional co-PI)
  - Ignacio Laguna
- Argonne National Laboratory
  - Marc Snir (Institutional co-PI)
  - Rinku Gupta
  - Sheng Di
- Targeted systems
  - Current systems at ORNL, LLNL and ANL, and the CORAL systems

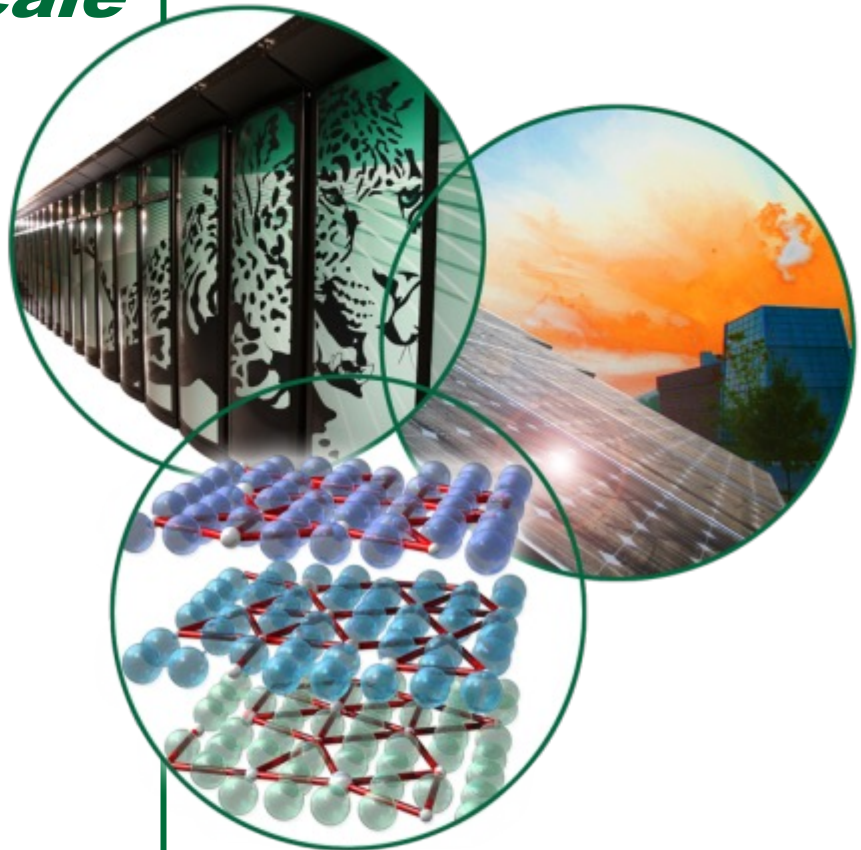
Questions?



C. Engelmann. Toward A Fault Model And Resilience Design Patterns For Extreme Scale Systems. Resilience Workshop 2015.

Resilience Design Patterns:

A Structured Approach to Resilience at Extreme Scale



Christian Engelmann
Oak Ridge National Laboratory

Early Career Research Program
Office of Advanced Scientific Research
Office of Science, US Department of Energy

U.S. DEPARTMENT OF ENERGY

OAK RIDGE NATIONAL LABORATORY
MANAGED BY UT-BATTELLE FOR THE DEPARTMENT OF ENERGY

Objectives

- This project offers a structured approach for resilient extreme-scale HPC systems
- Using a novel resilience design pattern concept, this project identifies and evaluates repeatedly occurring resilience problems and coordinates solutions throughout hardware and software components in HPC systems
- The results of this project enable the *systematic* improvement of resilience in extreme-scale HPC systems

C. Engelmann. Toward A Fault Model And Resilience Design Patterns For Extreme Scale Systems. Resilience Workshop 2015.

Current State of HPC Resilience

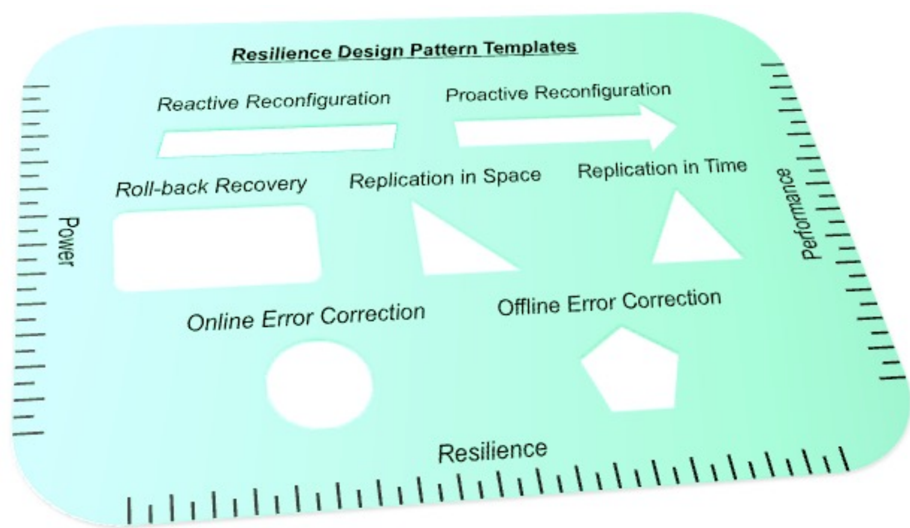
- Vendors have developed a number of hardware resilience technologies, including ECC/Chipkill and redundant components
- The HPC community has developed a number of software resilience technologies, including:
  - Application- and system-level checkpoint/restart
  - Fault-tolerant MPI and MPI message logging,
  - Redundant MPI and proactive fault tolerance,
  - Containment domains, and resilient solvers
- Application-level checkpoint/restart has been the predominant HPC fault tolerance method for decades
- Resilience is mostly measured by vendors with system MTTF and by users with application MTTF

Motivation

- There are no comprehensive evaluation methods & metrics that consider fault impact scope, handling coverage, and handling efficiency across the stack and the system
- There is also no clear understanding of protection against high-probability high-impact vs. less likely/harmful faults
- There is a general lack of coordination for resilience across the stack and the system to avoid costly overprotection
- There are no mechanisms and interfaces for coordination
- There is also no resilience portability across architectures

Approach

- Develop comprehensive methods and metrics to investigate and evaluate resilience in HPC systems
- Establish mechanisms and interfaces to facilitate coordination for resilience across the stack and the system
- Develop methodologies for optimizing the trade-off between performance, resilience, and power consumption at design time and runtime
- ***The approach centers on a novel resilience design pattern concept***



C. Engelmann. Toward A Fault Model And Resilience Design Patterns For Extreme Scale Systems. Resilience Workshop 2015.

Resilience Design Patterns

- Identify, evaluate, and coordinate repeatedly occurring resilience problems and solutions throughout the hardware/software stack and across system components
- Similar to parallel programming design patterns, a set of resilience design patterns covers the hardware and software architecture aspects of resilience in HPC systems
- Individual implementations of the same pattern may offer a different quality of service, i.e., performance, resilience, and power consumption
- The quality of service can be measured with the same metrics, based on the pattern scope for comparison and improvement

C. Engelmann. Toward A Fault Model And Resilience Design Patterns For Extreme Scale Systems. Resilience Workshop 2015.

Initial Resilience Design Patterns (1/3)

- Roll-back Recovery
  - A form of transaction processing: all variants of checkpoint/restart
  - *(1) state saving, (2) error and failure detection, and (3) state recovery*
- Replication in Space
  - Employs redundancy to mask faults : all variants of modular redundancy
  - *(1) input replication, (2) redundant execution in space, (3) failure detection and continued operation in degraded mode, and (4) output comparison and unification with error detection and correction*
- Replication in Time
  - Runs an entire application or parts of it twice (or more) to verify results
  - *(1) input replication, (2) redundant execution in time, (3) error and failure detection with potential re-execution, and (4) output comparison and unification with error detection and correction*

C. Engelmann. Toward A Fault Model And Resilience Design Patterns For Extreme Scale Systems. Resilience Workshop 2015.

OAK RIDGE National Laboratory

Initial Resilience Design Patterns (2/3)

- Online Error Correction
  - Employs data redundancy and forward error correction during execution
  - Examples include ECC, and resilient data representation and solvers
  - *(1) error detection and (2) error correction*
- Offline Error Correction
  - Employs data redundancy and/or domain-specific knowledge to correct errors after execution
  - Examples include resilient solvers with post-processing stage
  - *(1) error detection and (2) error correction*

Initial Resilience Design Patterns (3/3)

- Reactive Reconfiguration
  - Survive already activated faults through adaptation
  - Examples include dynamic load balancers and fault tolerant MPI
  - *(1) fault, error, and failure detection and (2) reconfiguration*
- Proactive Reconfiguration
  - Avoid the impact of imminent faults through anticipatory adaptation
  - Examples include preventative maintenance and process migration
  - *(1) fault detection and (2) reconfiguration component*

C. Engelmann. Toward A Fault Model And Resilience Design Patterns For Extreme Scale Systems. Resilience Workshop 2015.

Refining the Pattern Concept

- Refine the resilience design patterns concept and identify the specific patterns found in systems and applications
- Evaluate and model the performance, resilience, and power consumption of the identified patterns
- Design and implement interfaces and mechanisms to facilitate coordination between patterns for error and failure handling, including coverage coordination and error notifications

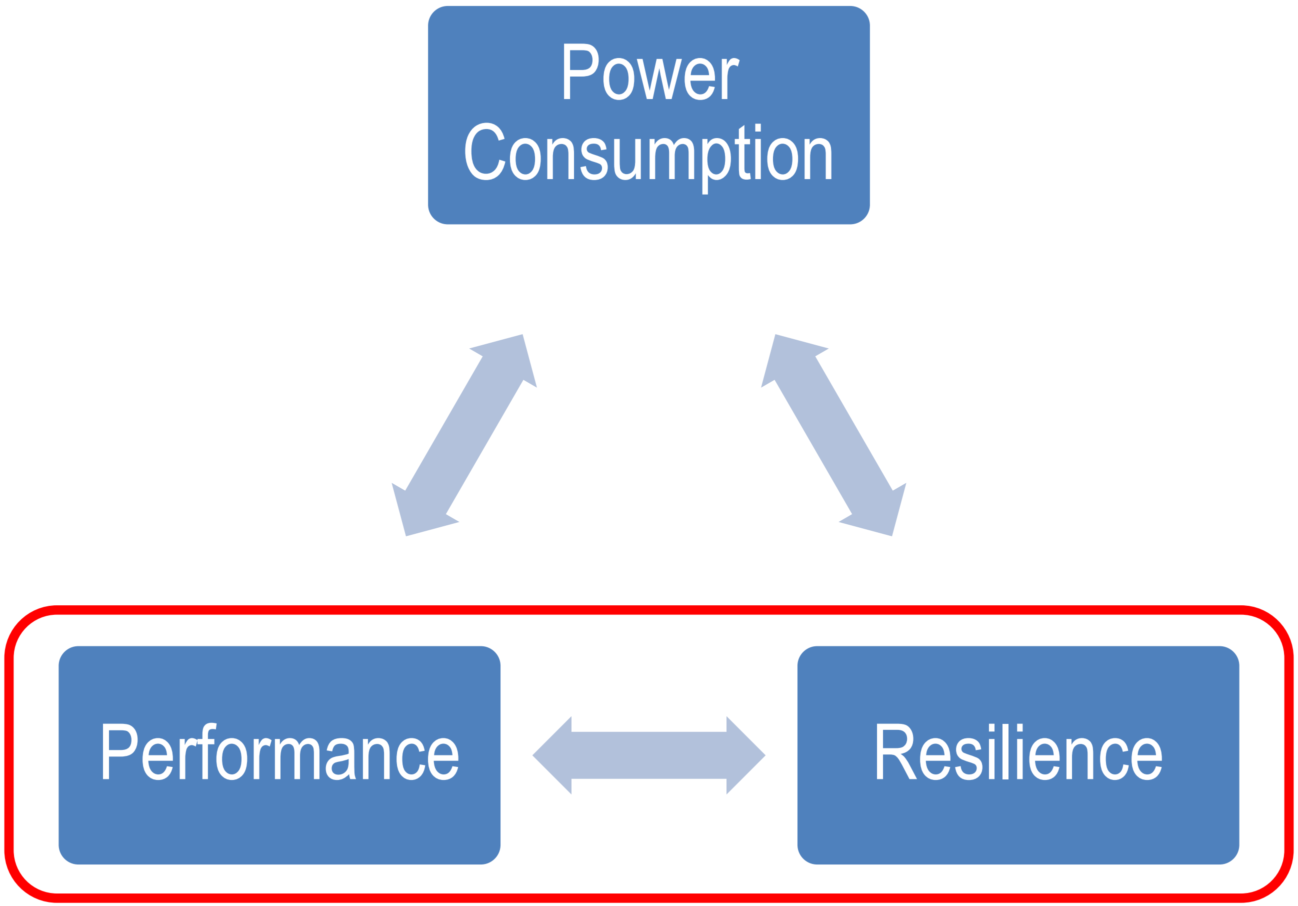
Employing the Pattern Concept

- Implement programming templates for resilience portability by leveraging the patterns, interfaces and mechanisms
- Develop a simulation tool and trade-off models for design space exploration by abstracting systems and applications as high-level resilience design pattern constructs
- Develop runtime trade-off models for tuning resilience solutions to handle errors and failures at the right level within the hardware/software stack and with the appropriate method

Leveraged Previous Accomplishments

- xSim - The Extreme-scale Simulator
- redMPI - A Redundant MPI
- Proactive Fault Tolerance Framework
- Hybrid Full/Incremental System-level Checkpointing
- Finject: A Fault Injection Framework
- Symmetric Active/Active High Availability for HPC System Services

HPC System Hardware/Software Co-Design: Optimizing Trade-offs

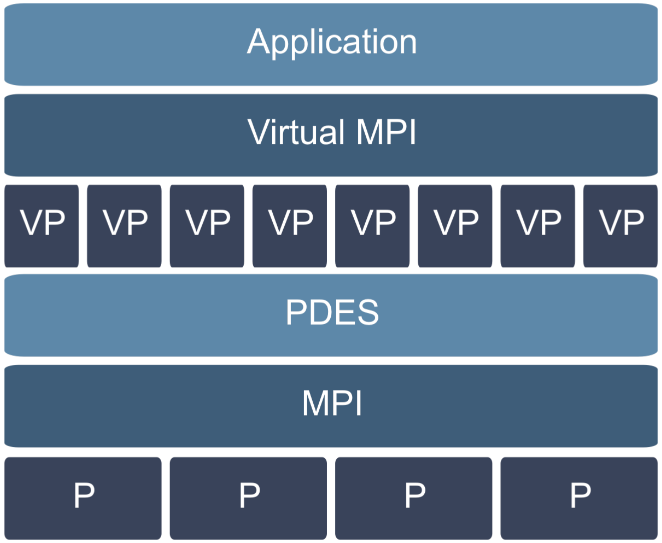


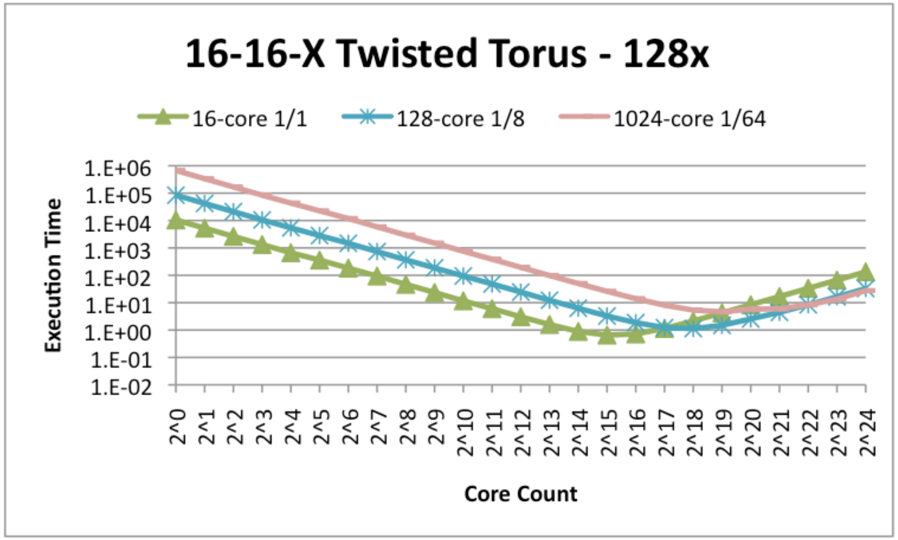
Examples: ECC memory, checkpoint storage, data redundancy, computational redundancy, algorithmic resilience

C. Engelmann. Toward A Fault Model And Resilience Design Patterns For Extreme Scale Systems. Resilience Workshop 2015.

xSim – The Extreme-Scale Simulator

- Combining highly oversubscribed execution, a virtual MPI, & a time-accurate PDES
  - Execution on native processor
  - Processor and network model
- Support for C/Fortran MPI
- Easy to use:
  - Compile with xSim header
  - Link with the xSim library
  - Execute: *mpirun -np <np> <application> -xsim-np <vp>*



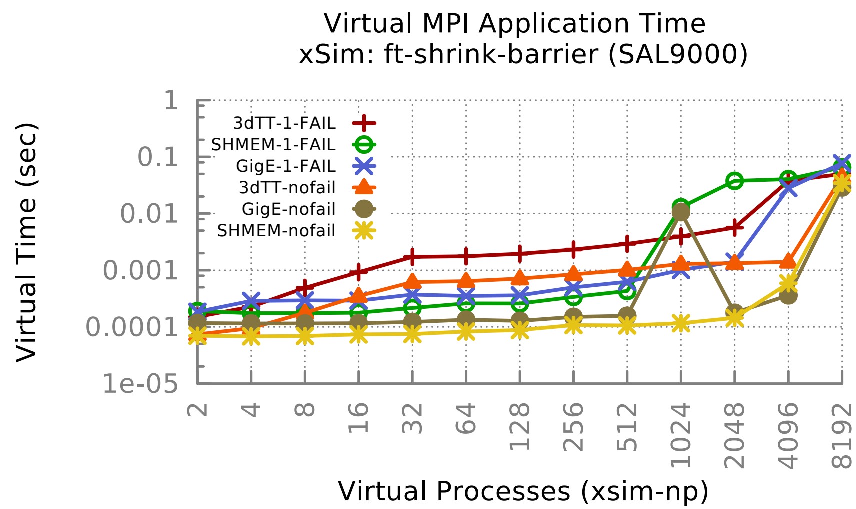


Resilience Simulation Features

- Simulated MPI process failure
  - Injection, propagation and detection in modeled architecture
- Simulated MPI application checkpoint, abort, and restart
  - Support for checkpoint/restart cycles until completion
- Simulated fault tolerant MPI
  - Support for resilient solvers using fault tolerant MPI (ULFM)
- Used in MCREX RX-Solvers project (Trilinos-based solver)

Table 1: Varying the checkpoint interval and system MTTF

$MTTF_S$	C	E_1	E_2	F	$MTTF_A$
—	1,000	5,248 s	—	0	—
6,000 s	500	5,258 s	7,957 s	1	3,978 s
6,000 s	250	6,377 s	7,074 s	1	3,537 s
6,000 s	125	6,601 s	6,750 s	1	3,375 s
3,000 s	500	5,258 s	10,584 s	2	3,528 s
3,000 s	250	6,377 s	8,618 s	2	2,872 s
3,000 s	125	6,601 s	7,948 s	2	2,649 s

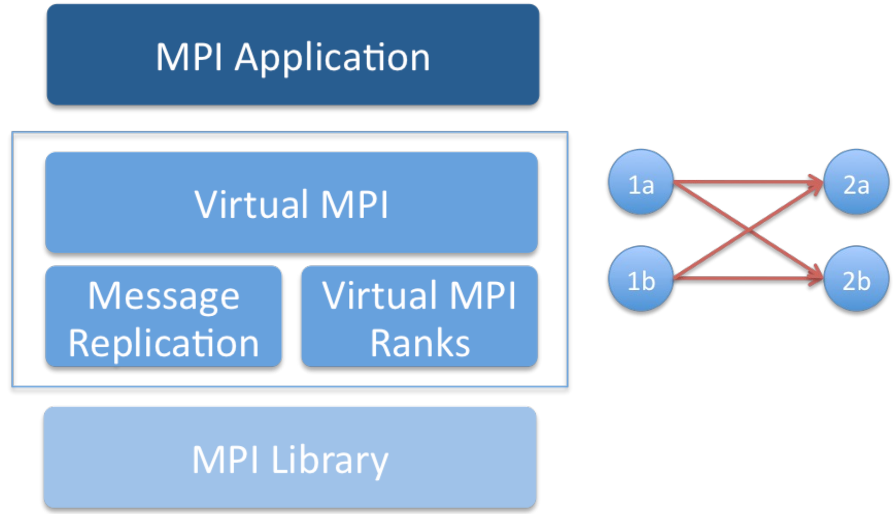


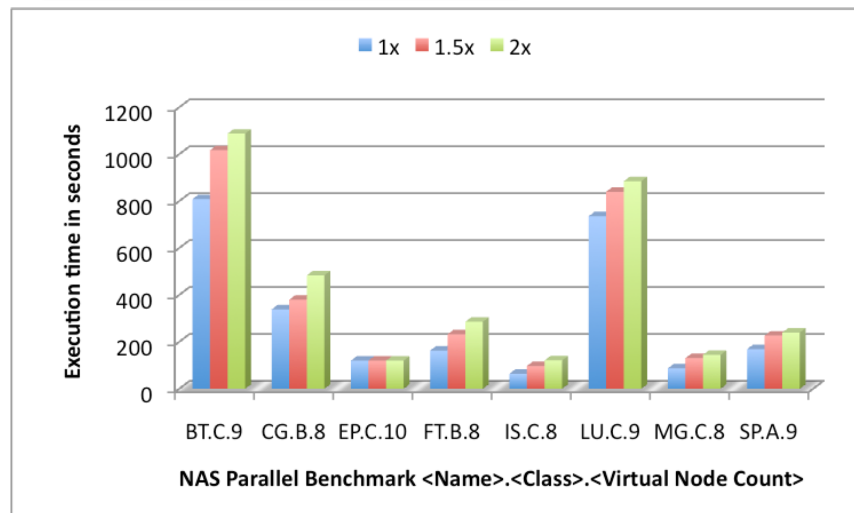
C. Engelmann. Toward A Fault Model And Resilience Design Patterns For Extreme Scale Systems. Resilience Workshop 2015.

OAK RIDGE National Laboratory

Process-level Redundancy atop MPI for Soft Error Resilience

- Transparent redundant execution of MPI apps. (redMPI)
- Interposition library between MPI and the app.
- App. runs with *r* * *m* ranks:
  - *r* ranks visible to the app.
  - *m* is the replication degree
- Fault model is fail-stop
- All messages are replicated
- File I/O is unified or replicated





OAK RIDGE National Laboratory | NC STATE UNIVERSITY | Sandia National Laboratories

C. Engelmann. Toward A Fault Model And Resilience Design Patterns For Extreme Scale Systems. Resilience Workshop 2015.

Using Redundancy for Soft Error Injection

- Study propagation of silent data corruption at runtime
- Taint one replica and use the other one as live control
- Disable error correction by the redundant MPI (redMPI)
- Compare MPI messages and record mismatch
- Obtain some sense of silent data corruption vulnerability
  - Corruption patterns
  - Propagation speed

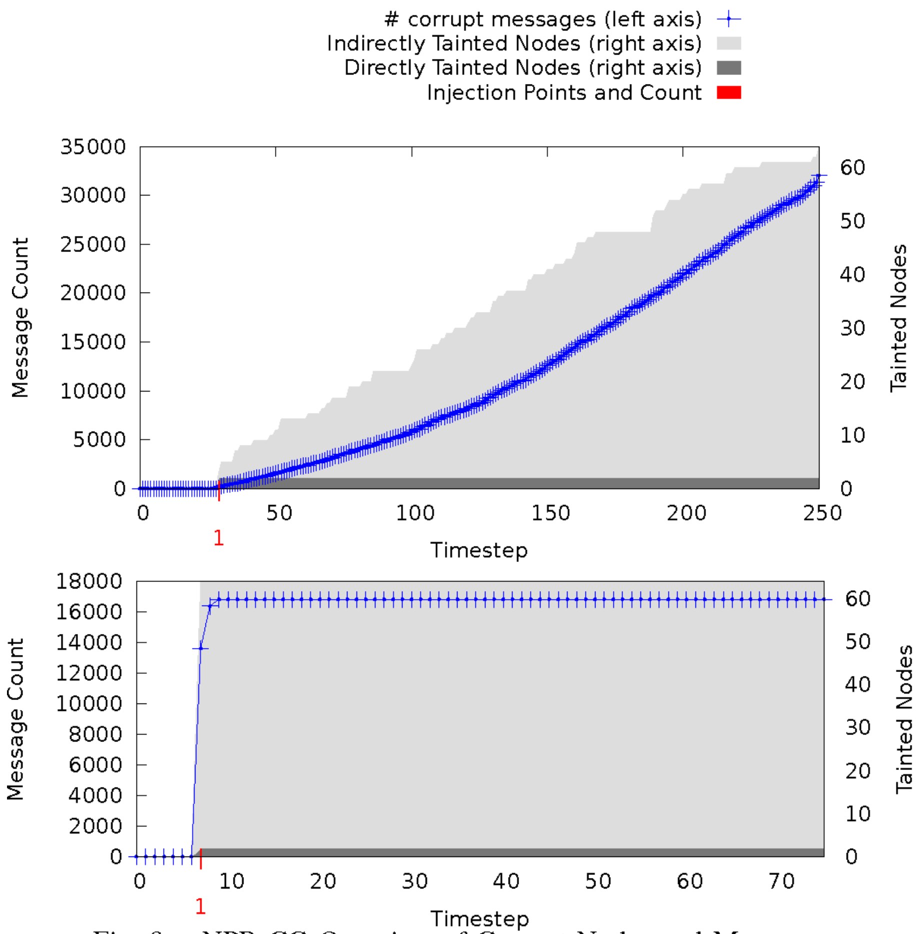


Fig. 8. NPB CG Overview of Corrupt Nodes and Messages

OAK RIDGE National Laboratory

NC STATE UNIVERSITY

Sandia National Laboratories

C. Engelmann. Toward A Fault Model And Resilience Design Patterns For Extreme Scale Systems. Resilience Workshop 2015.

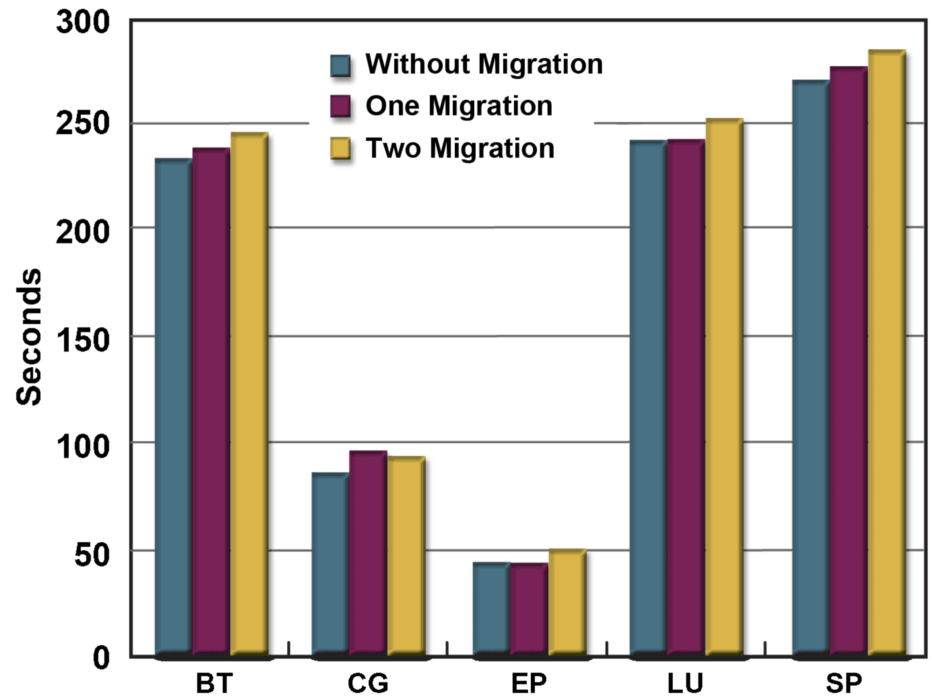
Proactive Fault Tolerance using Migration

- Relies on a feedback-loop control mechanism
  - Application health is constantly monitored and analyzed
  - Application is reallocated to avoid failures
  - Closed-loop control similar to dynamic load balancing
- Real-time control problem
  - Need to act in time to avoid imminent failures
- No 100% coverage
  - Not all failures can be anticipated

C. Engelmann. Toward A Fault Model And Resilience Design Patterns For Extreme Scale Systems. Resilience Workshop 2015.

VM-level Migration with Xen

- Single migration overhead
  - Live : 0.5-5.0%
- Double migration overhead
  - Live : 2.0-8.0%
- Migration duration
  - Stop & copy : 13-14s
  - Live : 14-24s
- Application downtime
  - Stop & copy > Live
- Node eviction time
  - Stop & copy < Live



NPB runs on 16-node dual-core dual-processor Linux cluster at NCSU with AMD Opteron and Gigabit Ethernet

Oak Ridge National Laboratory | NC STATE UNIVERSITY

C. Engelmann. Toward A Fault Model And Resilience Design Patterns For Extreme Scale Systems. Resilience Workshop 2015.

Process-Level Migration with BLCR

Single migration overhead

- Stop & copy : 0.09-6.00%
- Live : 0.08-2.98%

Single migration duration

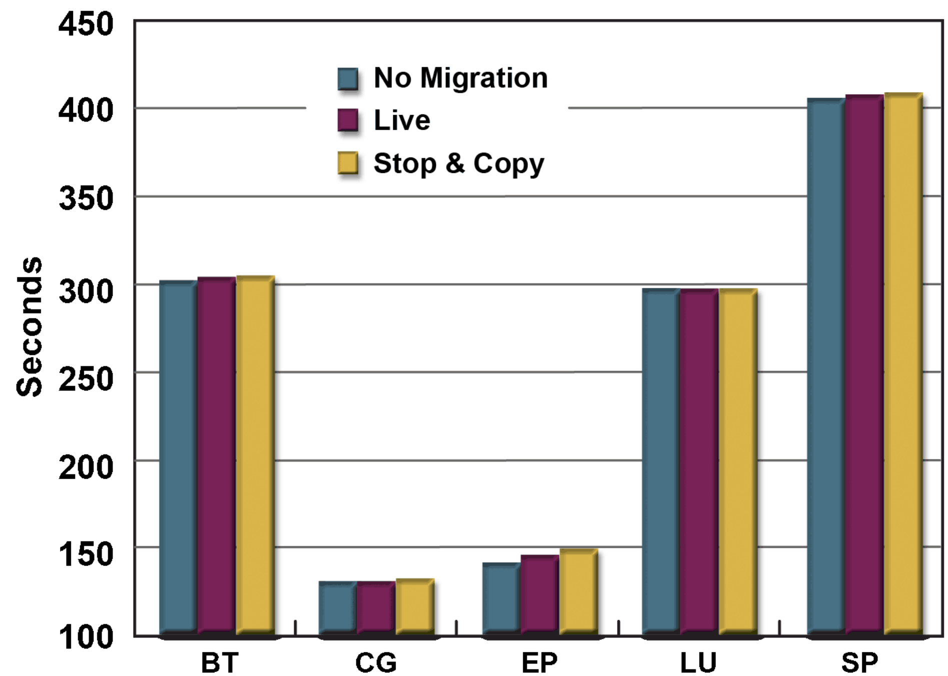
- Stop & copy : 1.0-1.9s
- Live : 2.6-6.5s

Application downtime

- Stop & copy > Live

Node eviction time

- Stop & copy < Live



NPB runs on 16-node dual-core dual-processor Linux cluster at NCSU with AMD Opteron and Gigabit Ethernet

Oak Ridge National Laboratory | NC STATE UNIVERSITY

Full/Incremental Checkpointing with BLCR

- Hybrid checkpointing:
  1 full and k incremental
  (part of BLCR distribution)
- Tracks dirty memory pages
- Full: Saves all pages
- Incremental: Appends dirty pages to checkpoint file
- Recovery: Scans file in reverse sequence
- Optimal at 1 full / 9 incr.

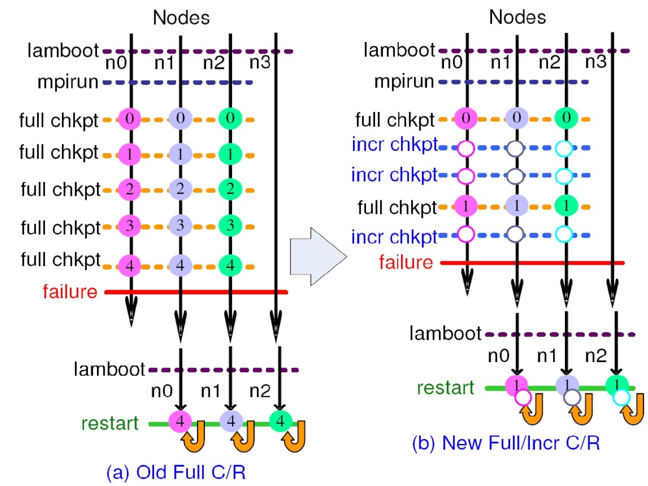
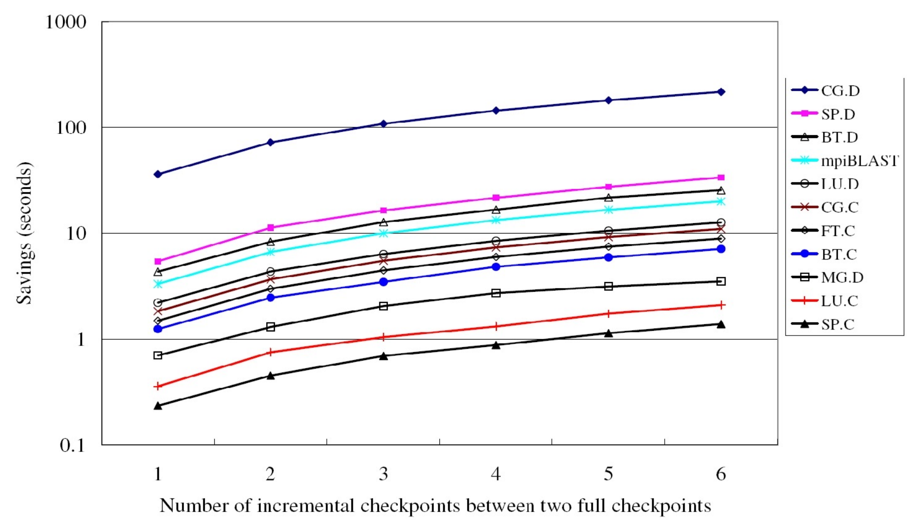


Fig. 1: Hybrid Full/Incremental C/R Mechanism vs. Full C/R



OAK RIDGE National Laboratory

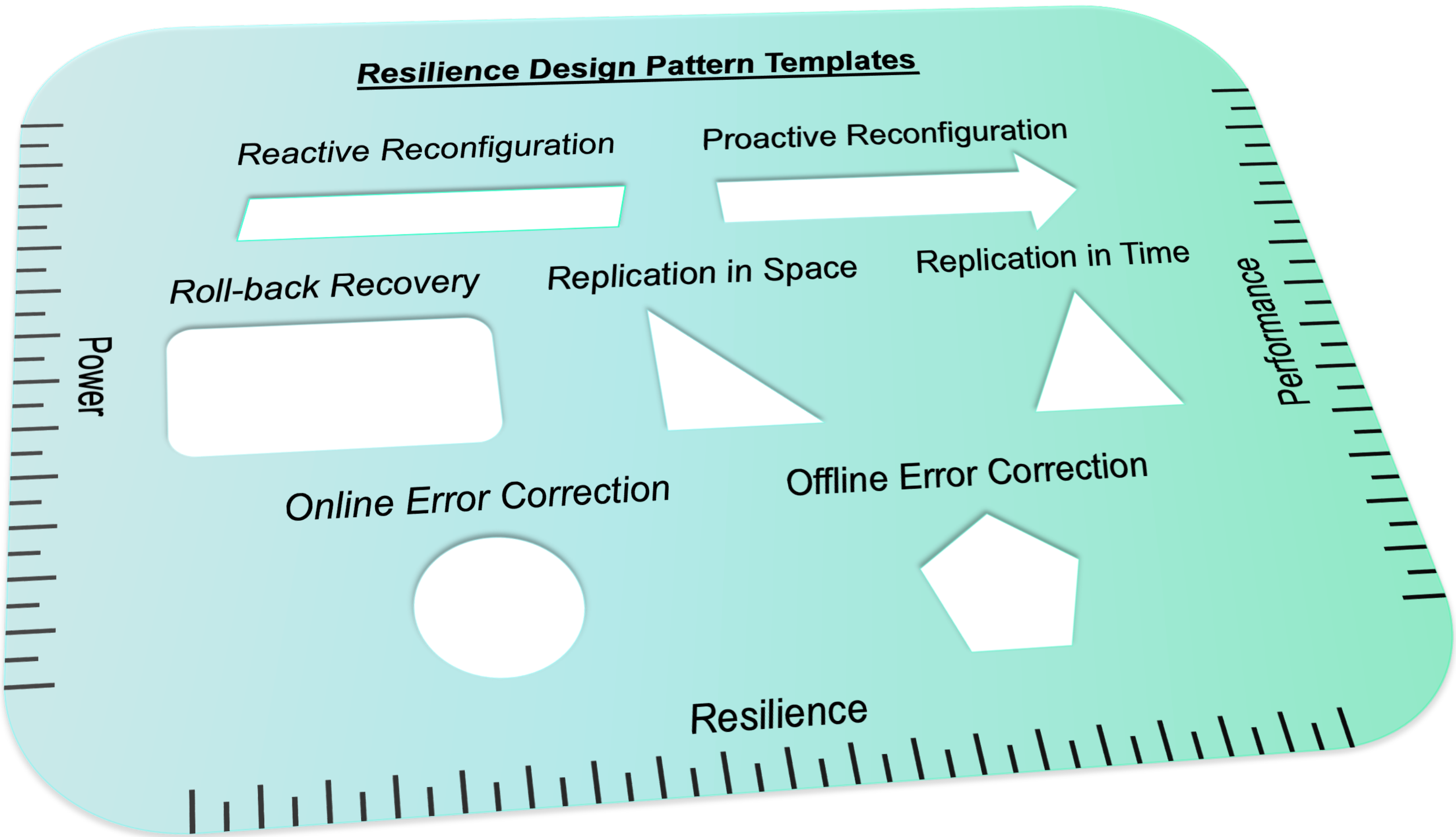
NC STATE UNIVERSITY

C. Engelmann. Toward A Fault Model And Resilience Design Patterns For Extreme Scale Systems. Resilience Workshop 2015.

Deliverables

- Year 1
  - Resilience design pattern specification documentation
- Year 2
  - Resilience design pattern specification with pattern models
- Year 3
  - Mid-term demo – Resilient solver with portable resilience
- Year 4
  - Design space exploration tool utilizing resilience design patterns
- Year 5
  - Final demo – exascale “swim lanes” through design space exploration

Questions?



C. Engelmann. Toward A Fault Model And Resilience Design Patterns For Extreme Scale Systems. Resilience Workshop 2015.