Motivation

• **The Paradox of Choice:**
  – Many possible solutions for resilience in extreme-scale high-performance computing systems [hardware, system software, algorithm-based, programming model-based, etc.]

  × Incomplete understanding of protection coverage against high probability & high impact vs. less likely & less harmful faults

  × No good evaluation methods & metrics that consider
    • Fault impact scope, handling coverage and handling efficiency
    • Performance, resilience and power trade-offs

  × No mechanisms and interfaces for coordination for avoidance of costly overprotection

  × No resilience portability across architectures and software environments
Design Patterns for Resilience

- A design pattern provides a generalizable solution to a recurring problem
- It formalizes a solution with an interface and a behavior specification
- Design patterns do not provide concrete solutions
- They capture the essential elements of solutions, permitting reuse and different implementations
- **State patterns** provide encapsulation of system state for resilience:
  - Persistent State, Dynamic State, Environment State and Stateless patterns
- **Behavioral patterns** provide encapsulation of detection, containment and mitigation techniques for resilience:
  - Strategy, Architecture, and Structural patterns
Anatomy of a Resilience Design Pattern

- A resilience design pattern is defined in an event-driven paradigm
- Instantiation of pattern behaviors may cover combinations of *detection*, *containment* and *mitigation* capabilities
- Enables writing patterns in consistent format to allow readers to quickly understand context and solution
• Taxonomy of resilience terms and metrics
• Survey of resilience techniques
• Classification of resilience design patterns
• Catalog of resilience design patterns
  – Uses a pattern language to describe solutions
  – 3 strategy patterns, 5 architectural patterns, 11 structural patterns, and 5 state patterns
• Case studies using the design patterns
• A resilience design spaces framework

Case Study: Checkpoint Recovery with Rollback

Diagram showing layers of software and hardware components.
Case Study: Proactive Process Migration
Case Study: Cross-Layer Hardware/Software Hybrid Solution

![Diagram showing the interaction between hardware and software layers in a hybrid solution.]

- **Hardware**
  - ECC Detection
  - DRAM
  - x86 or OpenPower
  - NVLink + IB
  - NVRAM
  - GPU or Accelerator
  - Local Storage & SAN

- **System Software**
  - Page-level Containment

- **Management**
  - File System
  - Numerical Libraries
    - Checksum Correction
  - Monitoring Framework
  - Job Scheduler
  - Scientific Domain Specific Libraries
  - MPI, OpenMP, CUDA Runtimes
  - Debugging, Profiling Libraries, Runtimes

- **Application Code**
Resilience Design Spaces Framework

- Design for resilience can be viewed as a series of refinements
- The design process is defined by 5 design spaces
- Navigating each design space progressively adds more detail to the overall design of the resilience solution
- A single solution may solve more than one resilience problem
- Multiple solutions often solve different resilience problems more efficiently
Design Space Exploration for Resilience

- Vertical and horizontal pattern compositions describe the resilience capabilities of a system.
- Pattern coordination leverages beneficial and avoids counterproductive interactions.
- Pattern composition optimizes the performance, resilience and power consumption trade-off.
Resilience Design Pattern Language

- Identifies relationships between patterns
  - Abstraction vs. specialization
  - Used with vs. conflict
  - Similarity
  - Domain
- Uses graph representation
- Enables structured analysis
Systematic Modeling & Design of Resilience Solutions

- Abstract the system with models:
  - System component models (performance, resilience and power consumption models)
  - Resilience design pattern models (performance, resilience and power consumption models)
  - Application models (performance, resilience and power consumption models)
- Evaluate solutions using modeling and simulation
- Discover suitability of pattern combinations for system-specific resilience problems
- Predict behavior on different hardware architectures and in different software environments
System Component Models

- Already extensive work by ORNL, ANL and LLNL in analyzing DOE systems
- The Catalog project identifies, categorizes and models the fault, error and failure properties of DOE systems
  - Fault, error and failure types
  - Probability distributions
  - Temporal and spatial locality and correlation
  - Propagation paths and detection latency

Fraction of each failure type on 5 ORNL systems

Failure inter-arrival time for 3 ORNL systems (MTBF as red vertical line)
Resilience Design Pattern Models

• Preliminary mathematical reliability and performance models for each pattern
  – Take into account detection latency and performance loss due to repair and/or system degradation

• Ongoing work in outcome-based metrics considers value and performance efficiency
  – Correctness and time to solution

• Preliminary power consumption models are still work in progress

<table>
<thead>
<tr>
<th>Pattern</th>
<th>Performance Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fault Diagnosis</td>
<td>( T_{system} = T_0 + \sum_{k=1}^{n} \frac{t_{inference}}{\eta} )</td>
</tr>
<tr>
<td>Reconfiguration</td>
<td>( T_{system} = T_{FF} + (1 - T_{FF}) \cdot \frac{n - 1}{n} + T_R )</td>
</tr>
<tr>
<td>Rollback</td>
<td>( T_{system} = (T_{FF} + \gamma) / \eta ) where ( T_{FF} = o + \delta / r )</td>
</tr>
<tr>
<td>Roll forward</td>
<td>( T_{system} = o + \delta / r )</td>
</tr>
<tr>
<td>Redundancy</td>
<td>( T_{system} = T_{SER} \cdot ((1 - \alpha) + \beta \cdot \alpha)) + T_{MV} )</td>
</tr>
</tbody>
</table>
Application Models

• Significant amount of existing work in application performance models

• Some amount of existing work in application reliability models
  – Application vulnerability studies
  – Error propagation patterns
  – Resilient solvers
  – More work is needed

• Some amount of existing work in application power consumption models
  – More work is needed
Modeling and Simulation for Design Space Exploration (Future Work)

- Model the performance, resilience, and power consumption of an entire system
- Start at compute-node granularity with
  - System component models
  - Resilience design pattern models
  - Application models
- Simulate dynamic interactions between the system, resilience solutions and applications
- Move to finer-grain resolution to include on-node communication, computation and storage
- Build upon prior work with the Extreme-scale Simulator (xSim)
Contact & Acknowledgements

• Web site: https://ornlwiki.atlassian.net/wiki/spaces/RDP

• PI: Christian Engelmann, engelmannnc@ornl.gov

• Work supported by the Early Career Program of the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research with program manager Lucy Nowell