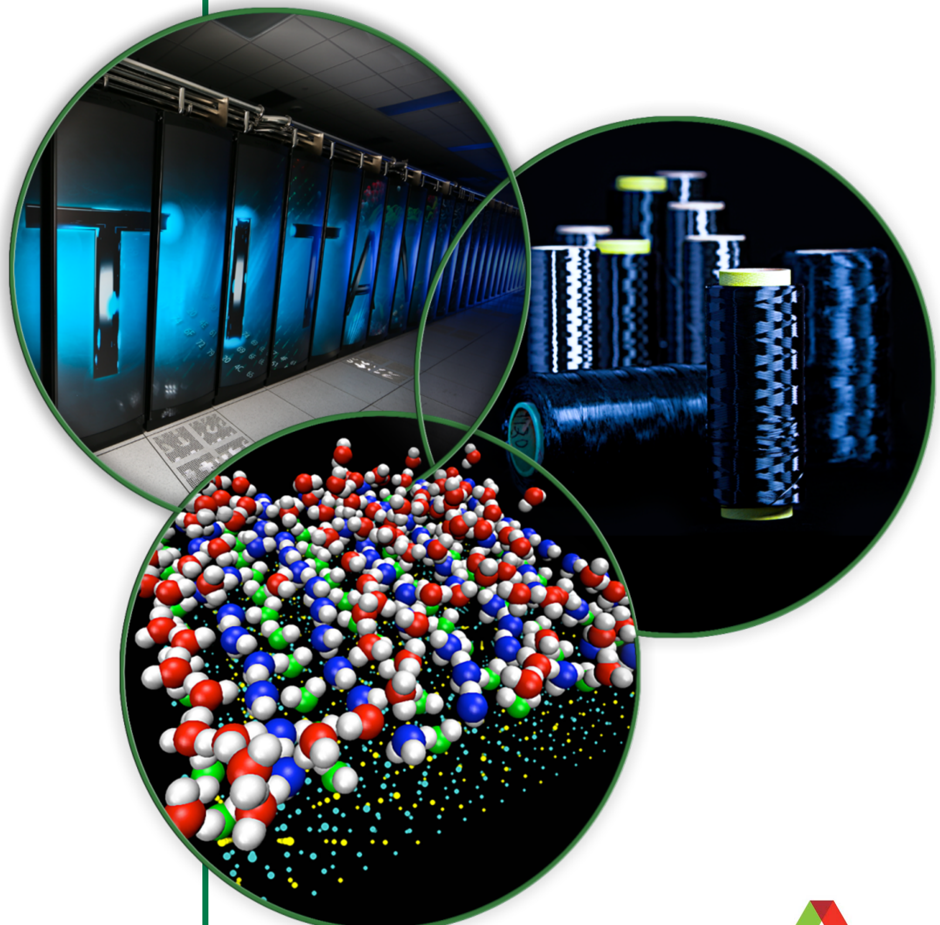
The Missing High-Performance Computing Fault Model

Christian Engelmann
Oak Ridge National Laboratory, USA

SIAM Conference on Parallel Processing for Scientific Computing (PP) 2016, Paris, France, April 12 -15, 2016.



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

Solving the Resilience Problem Requires Deep Understanding

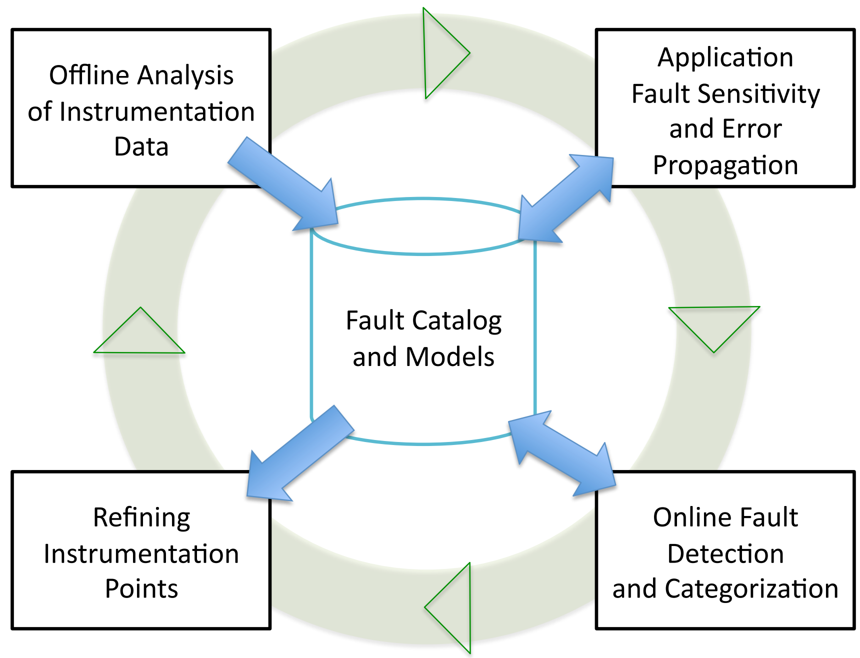
- **HPC resilience is a cost optimization problem**
  - Performance, resilience, power and deployment cost
- The challenge is to build a reliable system within a given cost budget that achieves the expected performance
- **This requires fully understanding the resilience problem** and offering efficient resilience mitigation technologies
  - **What is the fault model of HPC systems?**
  - **What is the impact of faults on HPC applications?**
  - How can mitigation in hardware and/or software help at what cost?

The Catalog Project

- The **Catalog project** creates the missing HPC fault model from fault data of actual large-scale production systems
  - A collaboration between **Oak Ridge National Laboratory**, **Argonne National Laboratory** and **Lawrence Livermore National Laboratory**
  - Funded by the **US Department of Energy (DOE)**
- The 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 extrapolates this knowledge to **exascale systems**
- This project will provide a **clear picture of the fault characteristics** in the DOE computing environments
- It will **improve resilience** through reliable fault detection at an early stage and actionable information for efficient mitigation

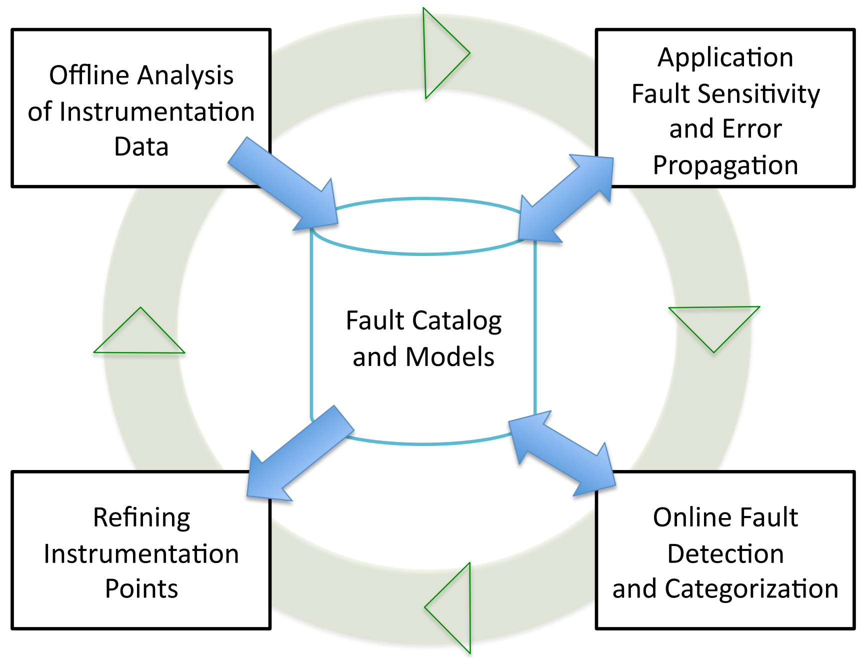
Approach (1/5)

- Create an **HPC fault catalog with**
  - A common **HPC fault taxonomy**
  - Specific **fault data** from systems
  - **Fault projections** of future systems
  - **System & component fault models**:
    - Representative models
    - Predictive models
    - Decision making models



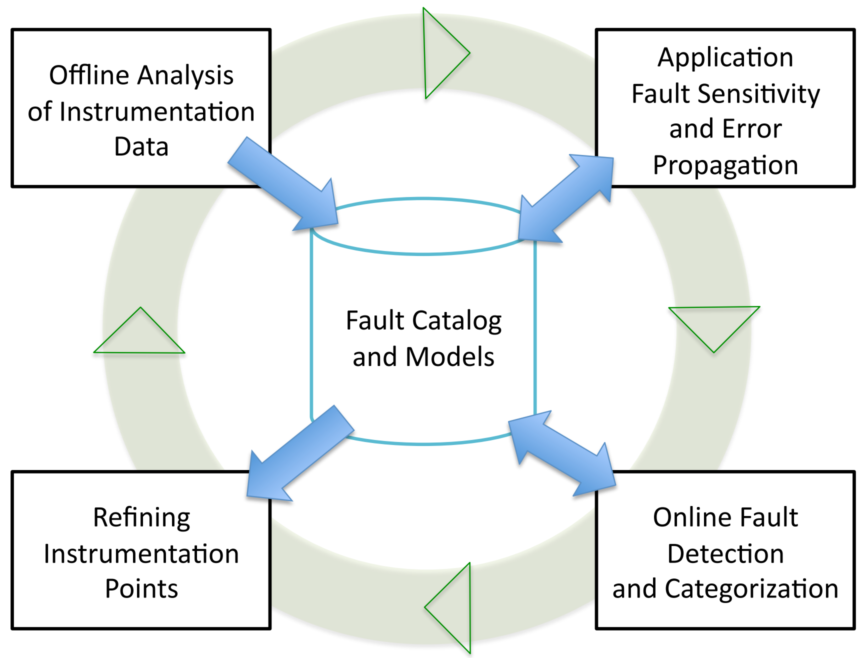
Approach (2/5)

- **Fuse system log data for <u>offline</u>**
  - Event identification
  - Event categorization
  - Root cause analyses
  - Event modeling
  - Visualization
- Leverages ORNL's RAVEN and ANL's HELO/ELSA tools



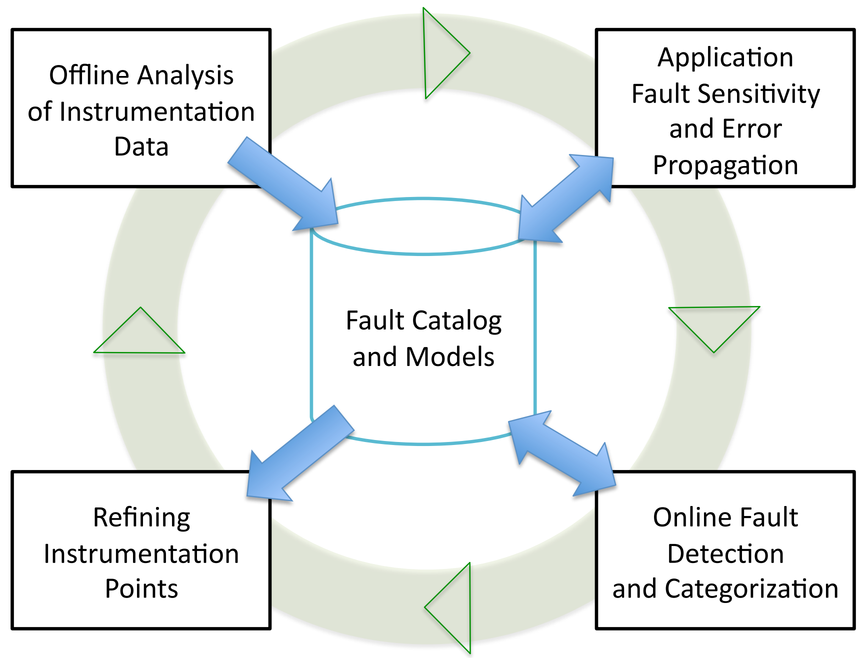
Approach (3/5)

- **Model the impact of faults on apps** using realistic fault injection
  - **Application vulnerability**, including algorithmic masking
  - **Error propagation** within applications, including error detection delays and containment
  - **Failure modes** of applications, including catastrophic, erroneous, and masked
- Using production codes, proxy apps and CORAL benchmarks
  - NEK5000/NEKbone, CoMD/ddcMD, Lulesh, AMG (hypre), Kripke
- Using LLNL's GREMLINs & ANL's FlipIt fault injection tools



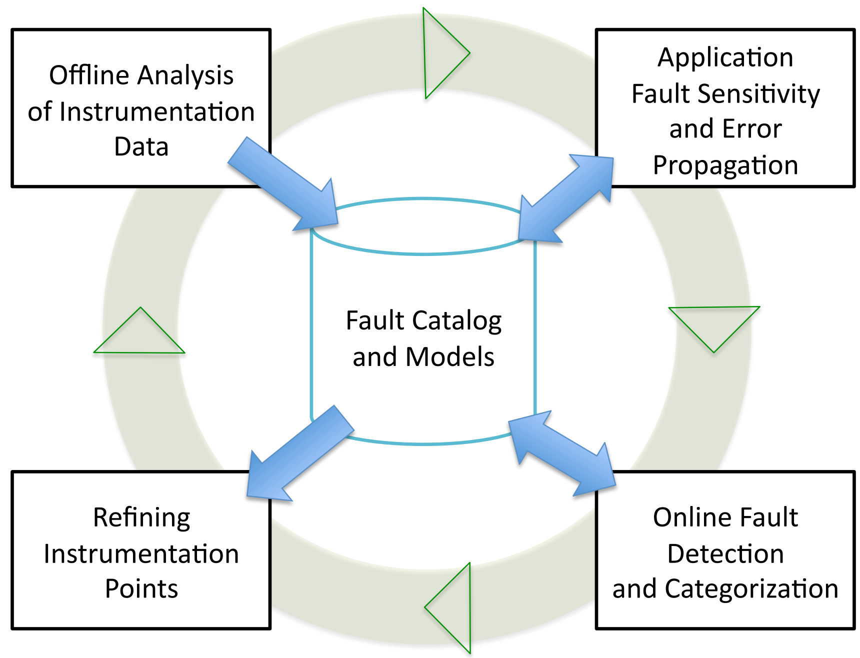
Approach (4/5)

- **Fuse real-time system health data for <u>online</u>**
  - Event identification
  - Event categorization
  - Root cause analyses
  - Event modeling
  - Visualization
- Based on ORNL's RAVEN and ANL's HELO/ELSA tools
- Uses system monitoring tools



Approach (5/5)

- **Identify additional instrumentation points** for
  - Early detection
  - More accurate categorization
- For example:
  - Instrument the file system MDS for feedback on transaction rates/delays



Team

- Oak Ridge National Laboratory
  - Christian Engelmann (PI)
  - Byung-Hoon (Hoony) Park
  - Devesh Tiwari
  - Saurabh Gupta (post-doc)
- Lawrence Livermore National Laboratory
  - Martin Schulz (Institutional co-PI)
  - Ignacio Laguna
- Argonne National Laboratory
  - Marc Snir / Franck Cappello (Institutional co-PI)
  - Rinku Gupta
  - Sheng Di
- Targeted systems
  - At **OLCF**, **ALCF** and **Livermore Computing**, including the CORAL systems

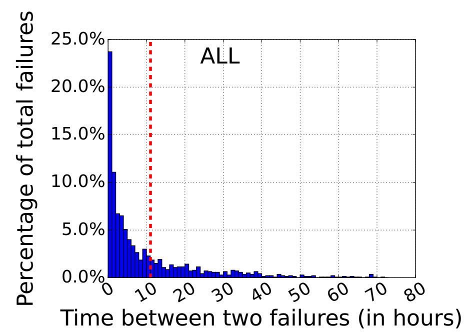
Status – Fault Catalog (1/2)

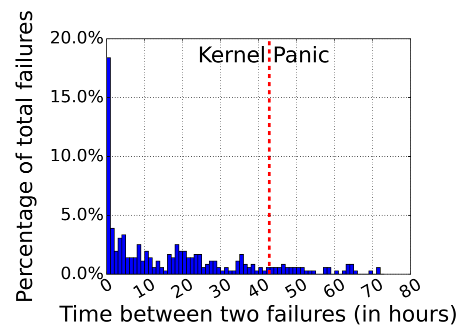
- Already analyzed a lot of data from Titan & Jaguar at ORNL
  - Node failures, memory errors, OS kernel panics, Lustre errors, etc.
- Titan had an initial lower reliability than Jaguar, but has improved
- Titan has long phases of stability with short phases of instability
  - Significant temporal & spatial locality
  - Partially invalidates common assumptions about MTBF in HPC
  - Significantly impacts checkpointing strategies and job queue lengths

System	Data Duration (days)	Failures	MTBF (hours)
Jaugar	1461	2620	13.38
Titan	900	1662	13.00

#	Jaguar Failure Types	User or System?
1	Error-> 'Out of Memory'	User
2	Error-> 'Machine Check Exception'	System
3	Error-> 'Node Heartbeat Fault'	System
4	Error-> 'Kernel Panic'	System
5	Error-> 'Link Inactive'	System
6	Error-> 'SeaStar Heartbeat Fault'	System

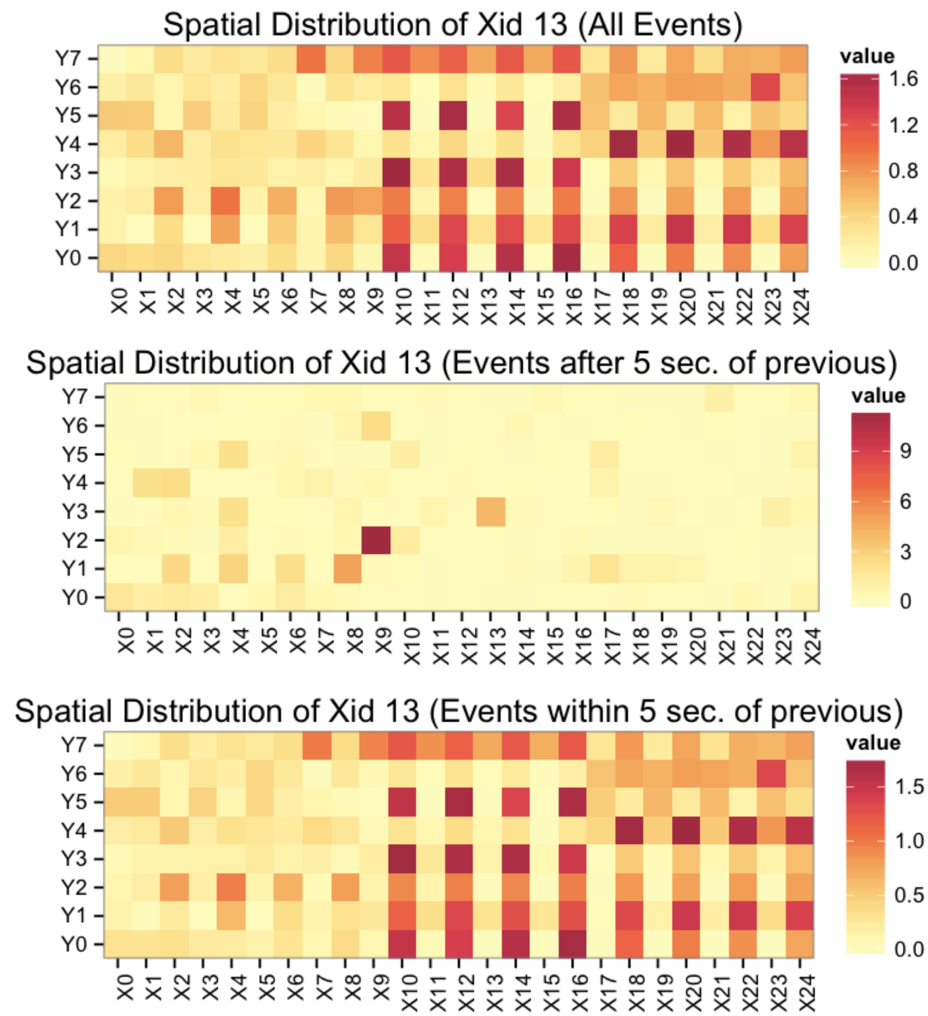
#	Titan Failure Types	User or System?
1	Type: Machine Check Exception	System
2	Type: Kernel Panic	System
3	Type: GPU DBE	System
4	Type: SXM Power Off	System
5	Type: Blade Heartbeat Fault	System





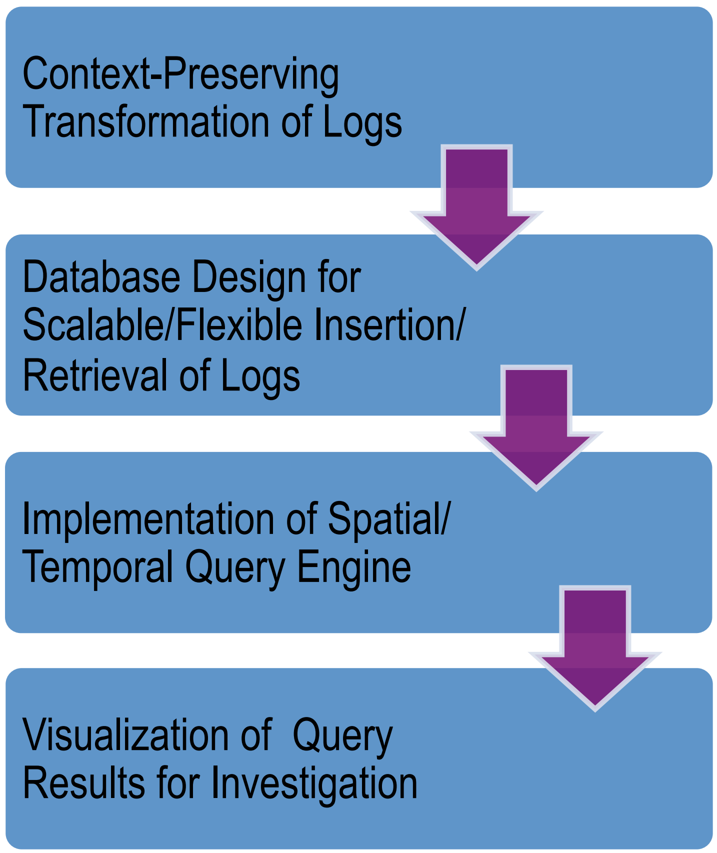
Status – Fault Catalog (2/2)

- Work on Titan GPU errors communicated to Nvidia
- Work on Titan DRAM & cache errors in collaboration with AMD
- Fault ≠ error ≠ failure
  - Working on root causes and propagation chains
- In the process of collecting similar data from Mira at ANL and from unclassified LLNL systems
- Getting ready to put together the first version of the catalog with Titan and Jaguar data
  - Taxonomy and fault statistics



Status – Offline Analysis

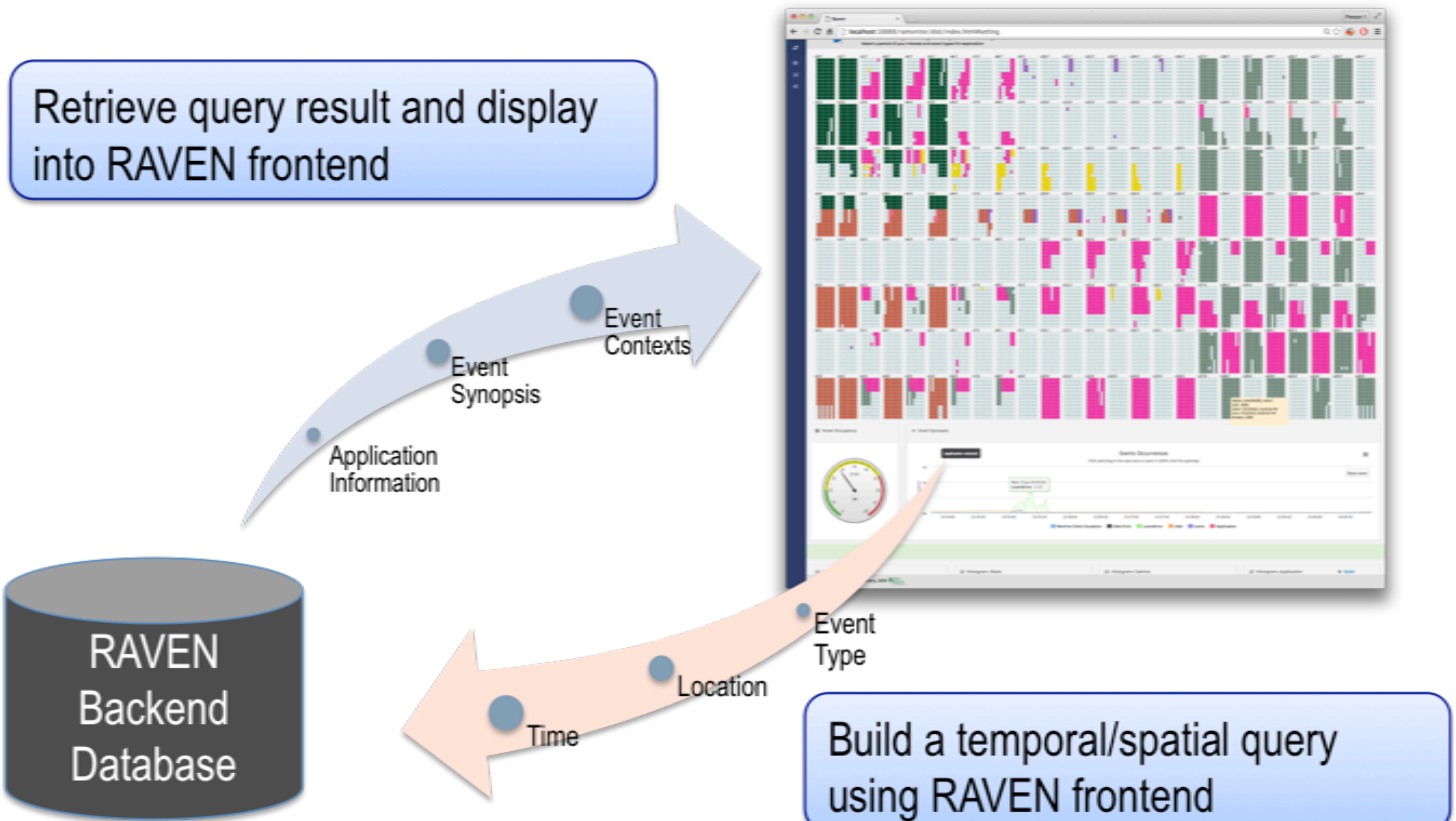
- Adapted ORNL's RAVEN offline analysis tool for Titan log data



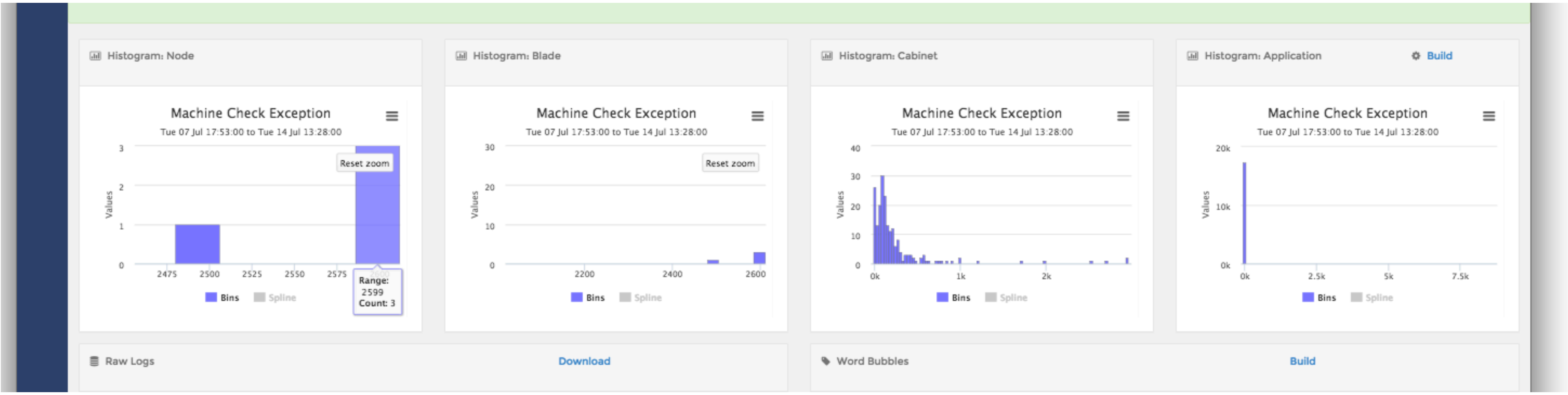
Adding new techniques in RAVEN:

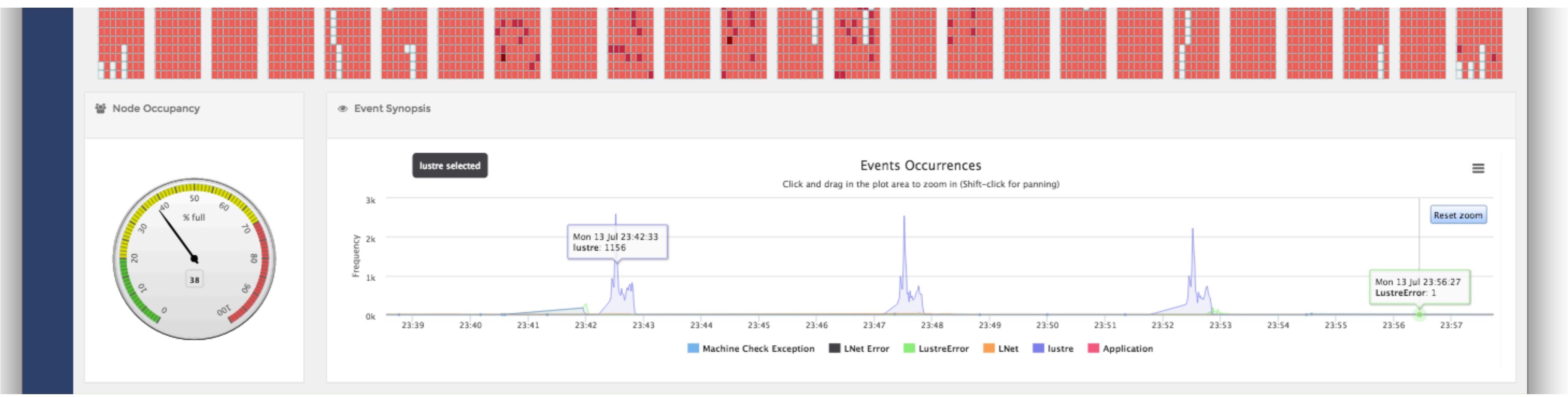
- Detection of emerging abnormal status of a supercomputer
- Taking an initial "big data" approach to analyzing unstructured logs
- Monitoring event streams to capture how much they are influencing one another to assess system health trajectory

RAS Data Analysis Through Visually Enhanced Navigation (RAVEN) (1/2)



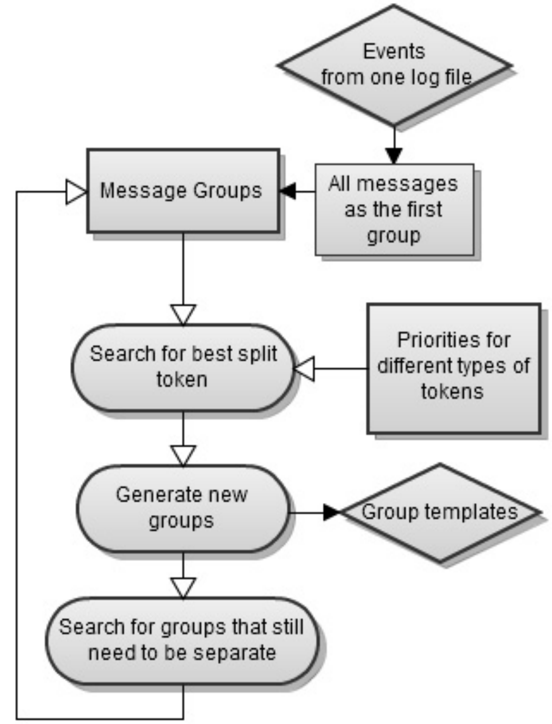
RAS Data Analysis Through Visually Enhanced Navigation (RAVEN) (2/2)

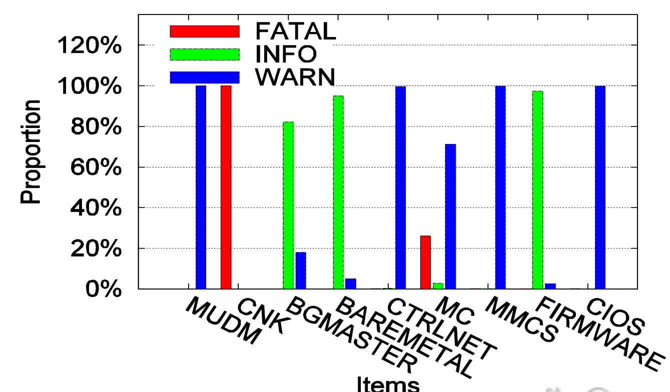




Hierarchical Event Log Organizer (HELO)

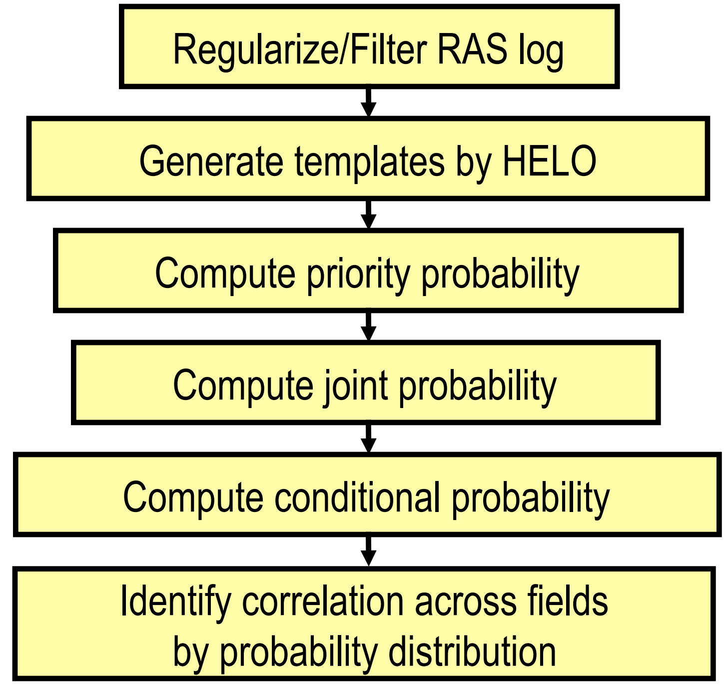
- HELO-based event log processing
  1. Extracts description fields for all events from a log
  2. Classifies all extracted descriptions based on syntax analysis
  3. Generates a template for each classified description group
  4. Inserts the description template ID back into the log
  - HELO has been develop for production use at NCSA
- Adapted HELO for Mira's RAS log
  - Analyzed a 1-month log
  - Generated 87 templates, i.e., identified 87 different event types

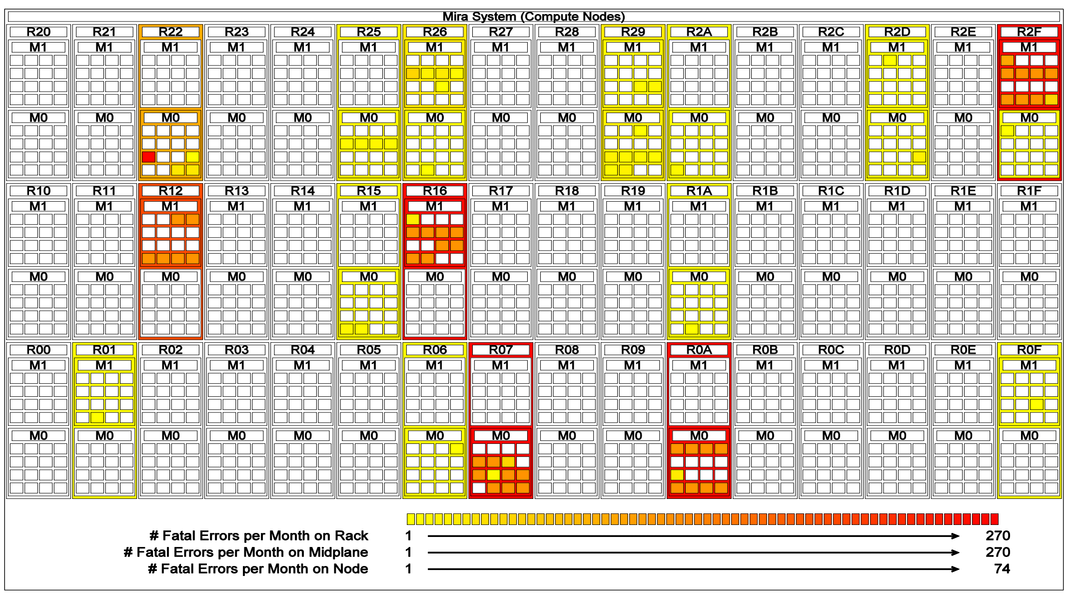




Correlation Analysis with HELO

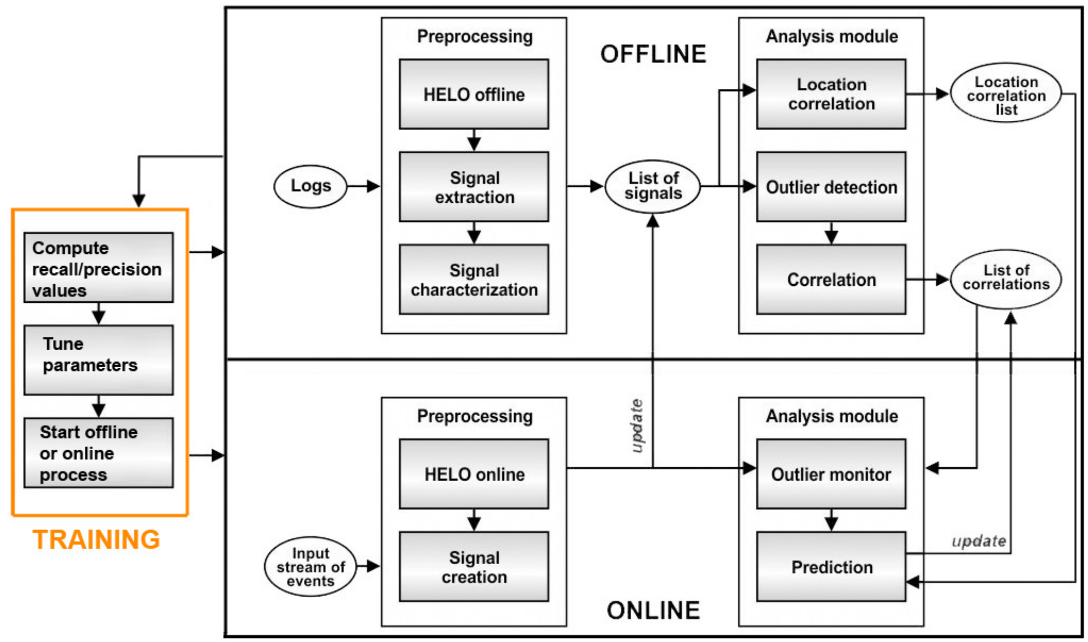
- Exploring across-field, temporal and spatial event correlation
- Employing spatial-temporal filtering for event statistics and root-cause analysis
  - RAS event vs. location
  - Job event vs. location





Event Log Signal Analyzer (ELSA)

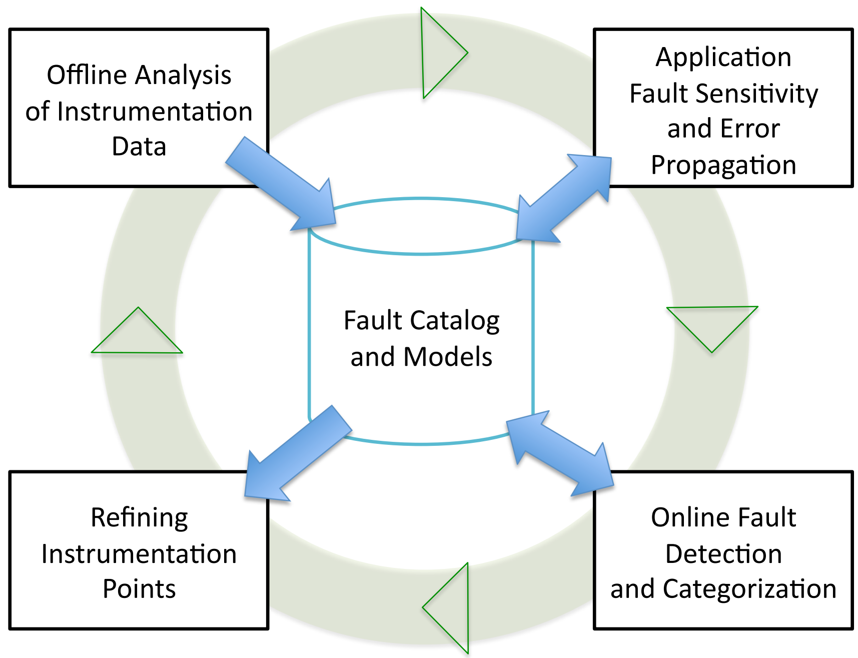
- Extracts correlation between events in HELO groups using signal analysis
- Transforms event groups into time series (based on time stamps)
- Analyzes the time series as signals to identify anomalies
- Correlates identified anomalies
- Adapting ELSA to Mira
- Plan to integrate HELO/ELSA with RAVEN for offline and online analysis



Deliverables

- 2016
  - **Initial fault catalog & models**
  - Comprehensive **offline analysis framework** with improved techniques
  - Infrastructure for realistic fault injection experiments
- 2017
  - Updated fault catalog & models
  - Characterization of application sensitivity using fault injection
  - **Refined instrumentation for offline analysis**
- 2018
  - **Final fault catalog & models**
  - **Application and system fault and error propagation models**
  - Comprehensive **online analysis framework** with real-time visualization

Questions?


Offline Analysis of Instrumentation Data
Application Fault Sensitivity and Error Propagation
Fault Catalog and Models
Refining Instrumentation Points
Online Fault Detection and Categorization