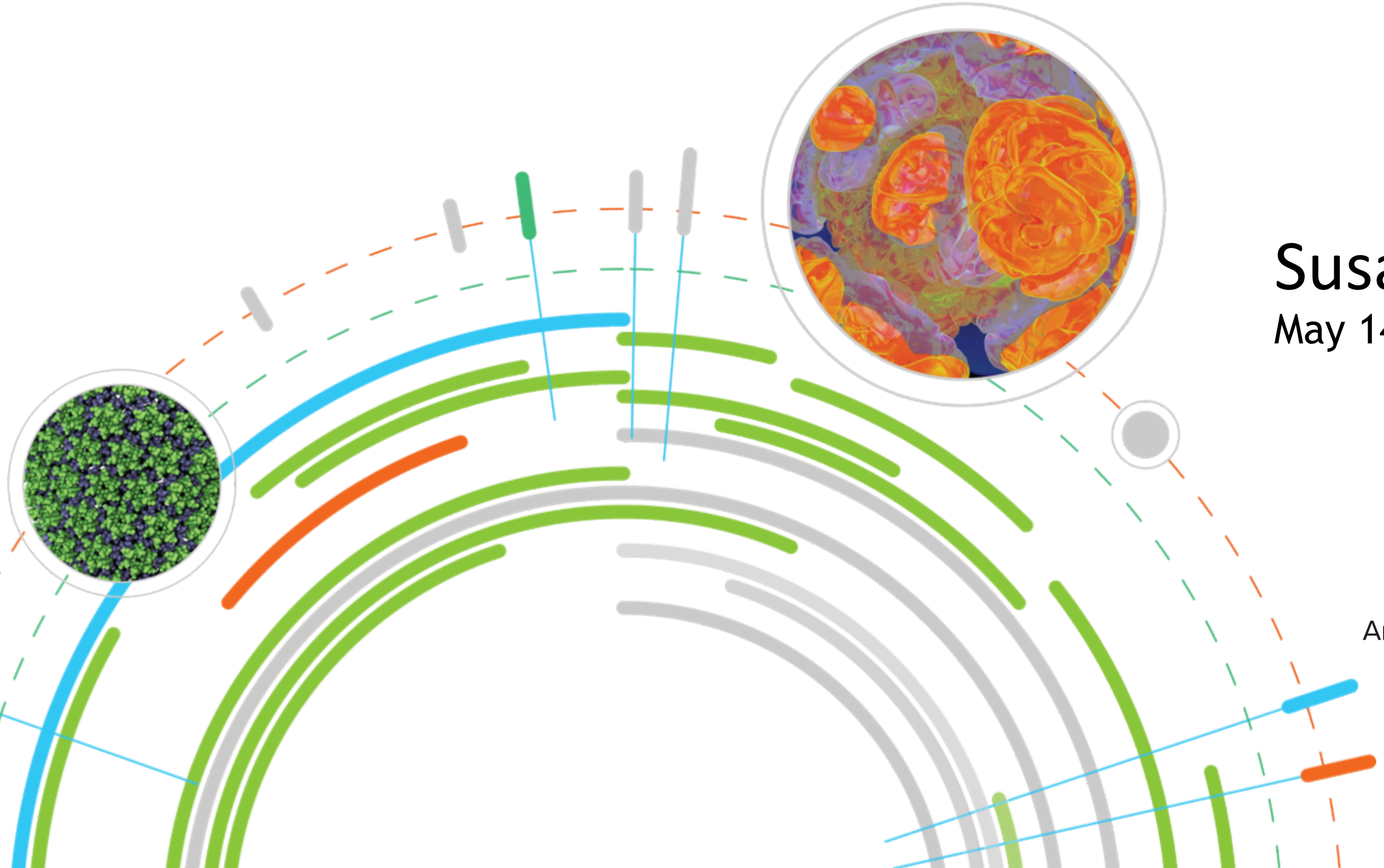
Continuous Improvement Using Comprehensive Root Cause Analysis



Susan Coghlan
May 14, 2015

Argonne Leadership Computing Facility

Argonne NATIONAL LABORATORY

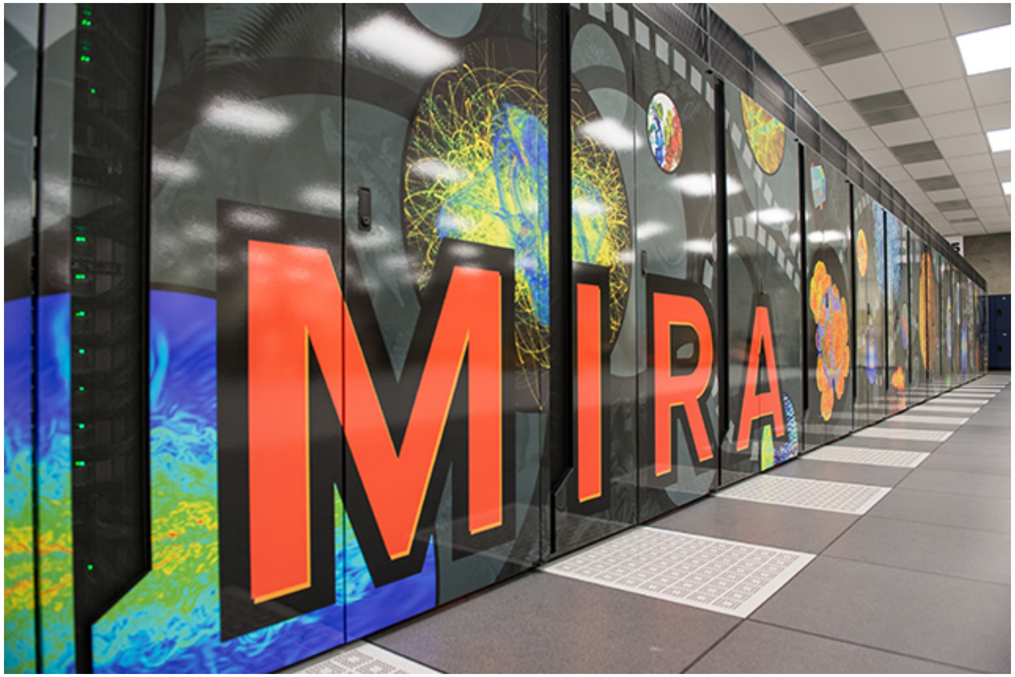
Argonne is Home to 5 National User Facilities

- Advanced Photon Source
- **Argonne Leadership Computing Facility**
- Argonne Tandem Linac Accelerator System
- Center for Nanoscale Materials
- Transportation Research and Analysis Computing Center
- Common characteristics
  - Scale
  - Cost
  - Uniqueness
  - Wide user base

What’s a Leadership Computing Facility?

- Open science for the world’s science community
- Two centers—ALCF at Argonne and OLCF at Oak Ridge National Laboratory
- Supported by DOE’s Advanced Scientific Computing Research Program
- Two architecturally diverse HPC resources
  - 10-100 times more powerful than systems typically available at other computer centers
- Primary mission: drive scientific and engineering breakthroughs
  - Small number of very large projects

Current Resources



Mira - *IBM Blue Gene/Q*

- 49,152 nodes / 786,432 cores
- 786 TB of memory
- Peak flop rate: 10 PF

Vesta - *IBM Blue Gene/Q*

- 2,048 nodes / 32,768 cores
- 32 TB of memory
- Peak flop rate: 419 TF

Cetus - *IBM Blue Gene/Q*

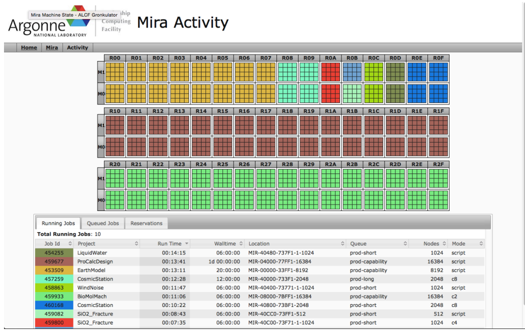
- 4,096 nodes / 65,536 cores
- 64 TB of memory
- Peak flop rate: 836 TF

Cooley - *Cray CS system*

- 126 nodes (each with 2 x Haswell 2.4 GHz 6-core CPUs and 1 x NVIDIA Telsa K80 GPU
- 47 TB memory
- Peak flop rate: 223 TF

Storage - Scratch: 28.8 PB raw capacity, 240 GB/s bw (GPFS); Home: 1.8 PB raw capacity; Tape: 16 PB of archival storage, 15,906 volume tape archive (HPSS)

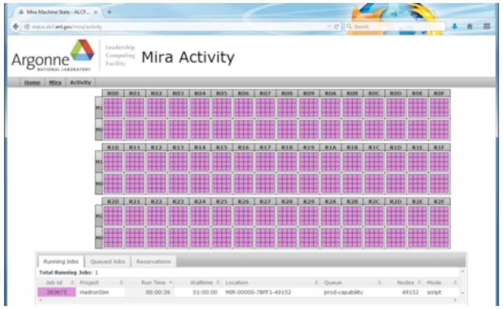
Leadership Computing Characteristics


Mira Machine State - ALCF Gronkulator
Argonne NATIONAL LABORATORY
Leadership Computing Facility
Mira Activity
Home
Mira
Activity
R00 R01 R02 R03 R04 R05 R06 R07 R08 R09 R0A R0B R0C R0D R0E R0F
R10 R11 R12 R13 R14 R15 R16 R17 R18 R19 R1A R1B R1C R1D R1E R1F
R20 R21 R22 R23 R24 R25 R26 R27 R28 R29 R2A R2B R2C R2D R2E R2F
M1
M0
Running Jobs
Queued Jobs
Reservations

Total Running Jobs: 10

Job Id	Project	Run Time	Walltime	Location	Queue	Nodes	Mode
454255	LiquidWater	00:14:15	06:00:00	MIR-40480-737F1-1-1024	prod-short	1024	script
459677	ProCalcDesign	00:13:41	1d 00:00:00	MIR-04000-77FF1-16384	prod-capability	16384	script
453509	EarthModel	00:13:11	20:00:00	MIR-00000-33FF1-8192	prod-capability	8192	script
457259	CosmicStation	00:12:28	12:00:00	MIR-40000-733F1-2048	prod-long	2048	c8
458863	WindNoise	00:11:47	06:00:00	MIR-40400-73771-1-1024	prod-short	1024	script
459933	BioMolMach	00:11:06	06:00:00	MIR-08000-7BFF1-16384	prod-capability	16384	c2
460168	CosmicStation	00:10:22	06:00:00	MIR-40800-73BF1-2048	prod-short	2048	c8
459082	SiO2_Fracture	00:08:43	06:00:00	MIR-40CC0-73FF1-512	prod-short	512	script
459800	SiO2_Fracture	00:07:35	06:00:00	MIR-40C00-73F71-1-1024	prod-short	1024	c4

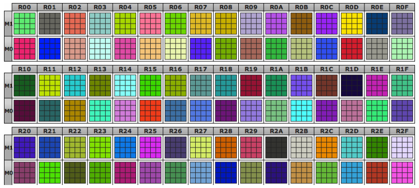
- ⊙ Capability is core to the LCF mission
  - ⊚ Scheduling policy encourages large, long jobs
  - ⊚ Smallest job allowed - 512 nodes (8k cores, 32k threads)
  - ⊚ Maximum # of jobs at any point in time is 96
  - ⊚ Averages around 200 jobs per day
  - ⊚ Sometimes one job running across full system for many hours – 49,152 nodes (786k cores, 3.1M threads)


Mira Machine State - ALCF...
status.alcf.anl.gov/mira/activity
Search
Argonne NATIONAL LABORATORY
Leadership Computing Facility
Mira Activity
Home
Mira
Activity
R00 R01 R02 R03 R04 R05 R06 R07 R08 R09 R0A R0B R0C R0D R0E R0F
R10 R11 R12 R13 R14 R15 R16 R17 R18 R19 R1A R1B R1C R1D R1E R1F
R20 R21 R22 R23 R24 R25 R26 R27 R28 R29 R2A R2B R2C R2D R2E R2F
M1
M0
Running Jobs
Queued Jobs
Reservations

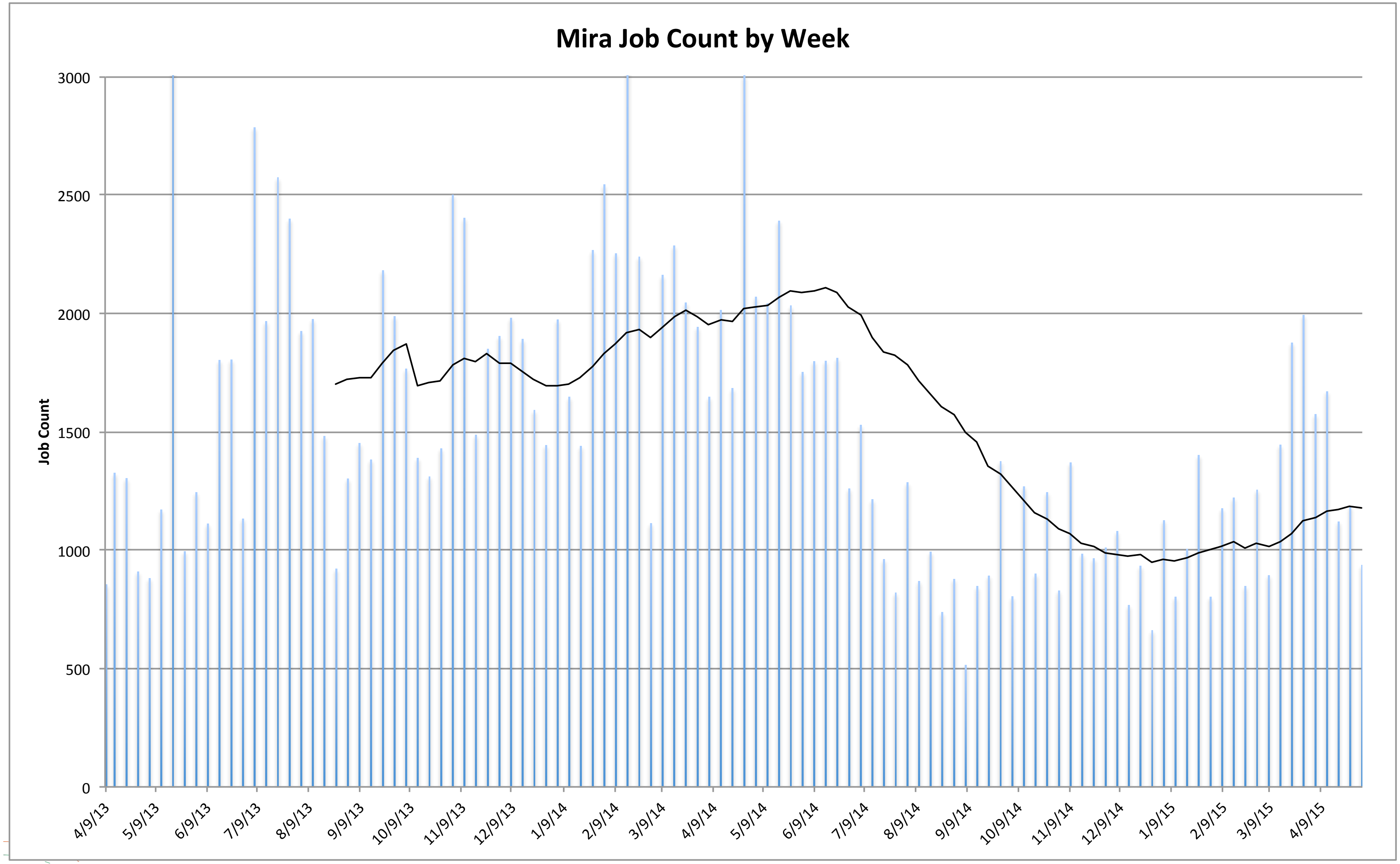
Total Running Jobs: 1

Job Id	Project	Run Time	Walltime	Location	Queue	Nodes	Mode
393675	HadronSim	00:00:39	01:00:00	MIR-00000-7BFF1-49152	prod-capability	49152	script

- ⊙ Applications requirements are different
  - ⊚ Fast low-latency communication required
  - ⊚ No jitter for nodes, slowest node == speed for all nodes
  - ⊚ Reproducibility for both performance and results required
  - ⊚ Parallel runtime environment is not fault-tolerant, recovery is typically with checkpoint/restart
- ⊙ Small (relatively) number of jobs and importance of each is integral


R00 R01 R02 R03 R04 R05 R06 R07 R08 R09 R0A R0B R0C R0D R0E R0F
R10 R11 R12 R13 R14 R15 R16 R17 R18 R19 R1A R1B R1C R1D R1E R1F
R20 R21 R22 R23 R24 R25 R26 R27 R28 R29 R2A R2B R2C R2D R2E R2F
M1
M0

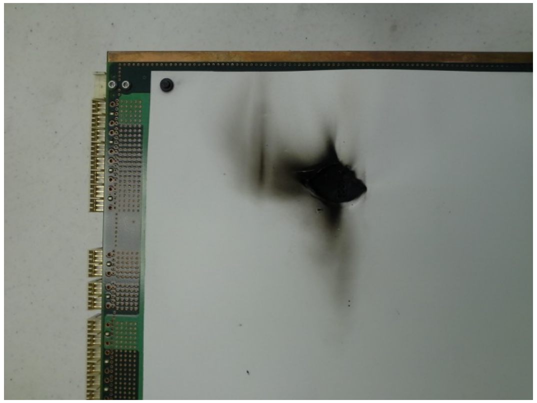
Mira Job Count by Week



Argonne Leadership Computing Facility

In the beginning…

- ALCF founded (in real life) in 2007
- Started from scratch, including building a data center
  - No data center on campus capable of supporting power (2MW), cooling (>220K CFM air flow), space requirements (6,000 sq ft)
- First large production resource (Intrepid) deployed in 2008/2009
  - IBM Blue Gene/P 500 TF, debut at #3 on Top 500 List
- Major challenges typical of these tightly coupled, complex, first of their kind, extreme scale supercomputers
  - Intermittent incorrect answers - replacement of almost all nodes, twice
  - Power supplies popping - redesign and replacement of all BPMs
- Priorities
  - Hire staff
  - Commission data center
  - Deploy hardware
  - Get correct answers and stable enough systems
  - Get users on and doing science



DOE reporting requirements added

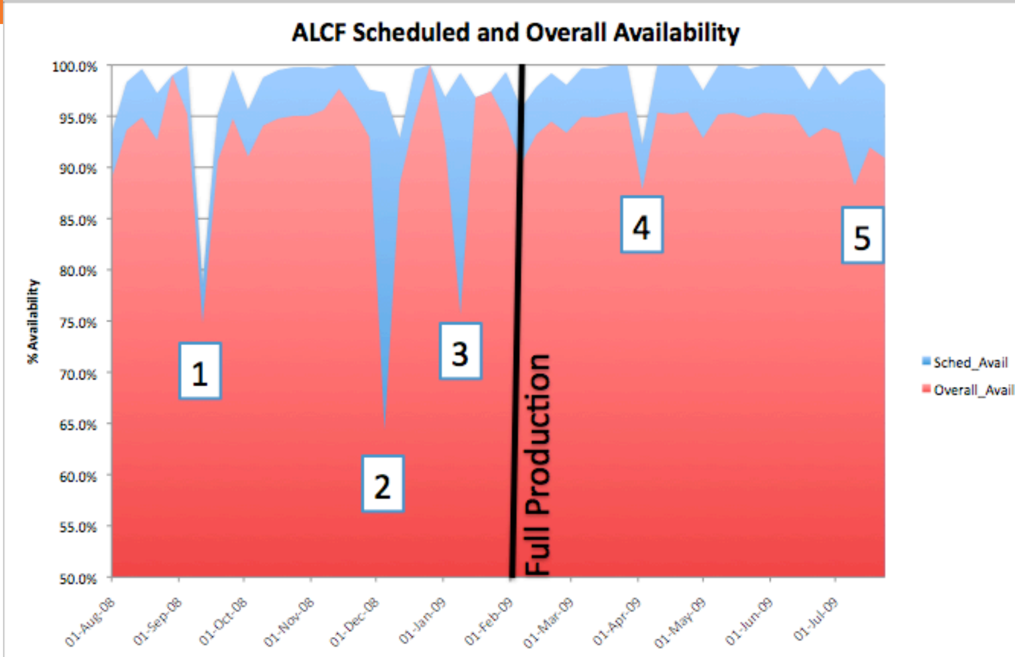
- Summer of 2009, DOE asked for Operational Assessment Report (OAR)
  - DOE's report to US Office of Management & Budget
  - Requirement to report on availability, utilization, MTTI/MTTF, etc.
- No explicit tracking of necessary data
- Blue Gene (BG) control system auto-gathers lots of data
  - Job data – multiple records for every job
  - Parts inventory and history for every HW component in system
  - RAS events – all info, warn, fatal
  - Environmental data from all components – voltage, current, temp, etc.
- Plus specialized and standard system logs
- Too much data from many sources
  - ~100M records/year for BG database alone
  - Difficult to manually calculate required metric actuals for DOE OAR

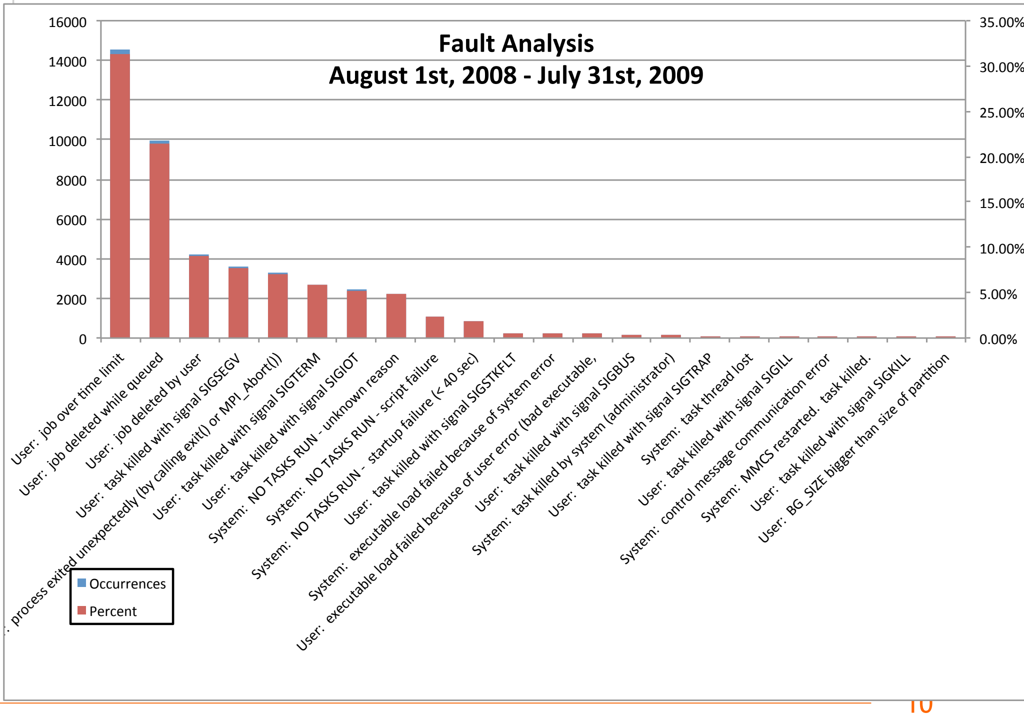
Automated Failure Analysis (AFA) Project

- Goal: Gather data, build list of system interrupts and job failures, categorize as User, System, Unknown and by component to assist with calculating number for OAR
- Data sources:
  - Blue Gene control system database
  - GPFS logs
  - Resource manager logs
  - MMCS (including boot) logs
  - Job stdout/stderr files
- Series of programs run by a shell script
  - Perl, Python, SQL, bash
- Analyzed all failed jobs and system failures
  - Correlated jobs to system SW/HW failures using time, messages, and location matching
  - Categorized all system interrupts by component that failed
  - Categorized all job failures as User, System, or Unknown
- Run once for full reporting period, dumped out CSV files
- Final step was to manually process CSV files using MS Excel

First OAR Report

- We got the necessary numbers
  - Overall Availability: 92.1%
  - Scheduled Availability: 97.5%
  - Utilization: 65.3%
  - 5 Major outages noted
  - 36.3% jobs failed
  - 10.4% failed marked System
- But there were issues
  - Single point in time, output from process not fed back into data
  - AFA good start, not complete story
  - Manual analysis plagued by errors, not consistent, built from staff memory of long past events



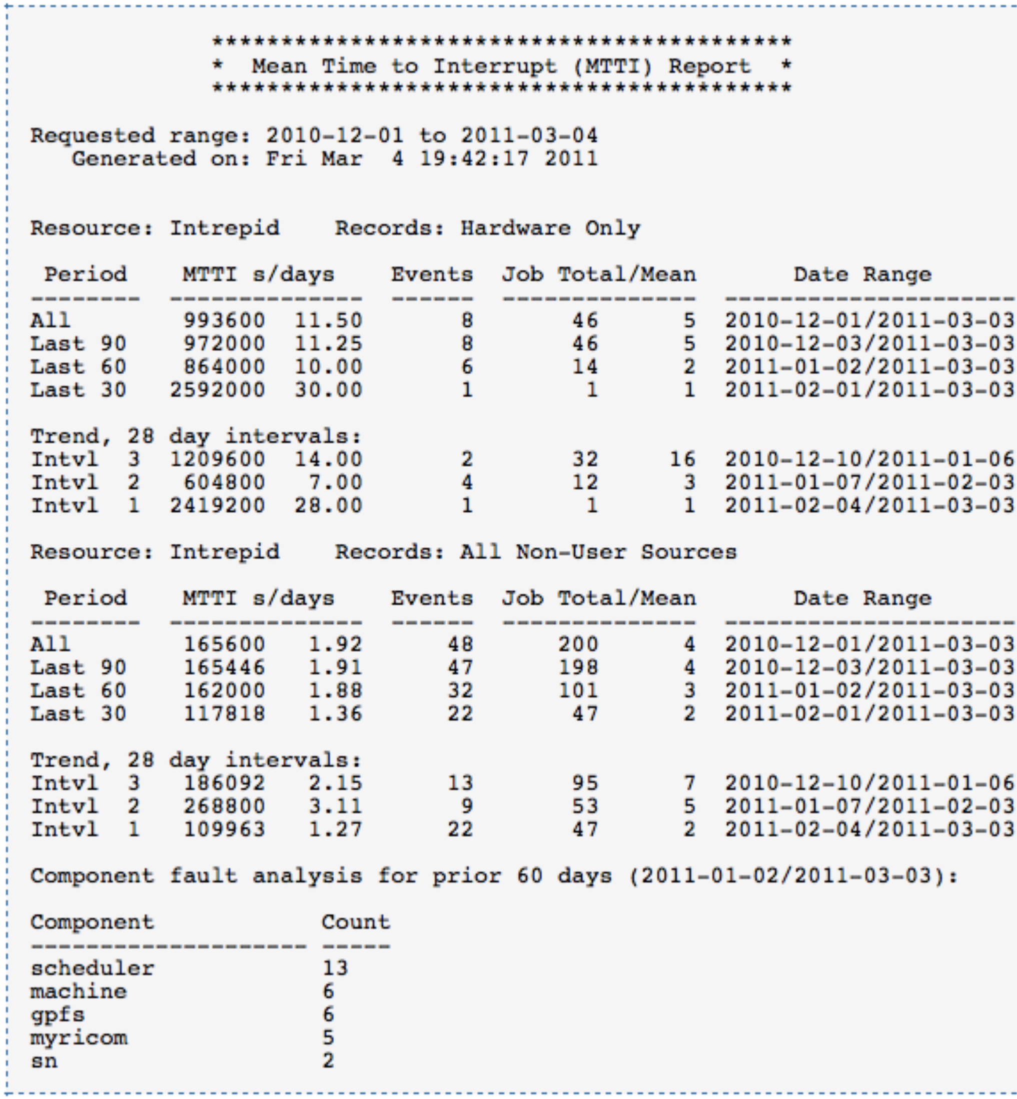


OARTool and OARdb Project

- Goal: Provide central repository for availability/interrupt data and tools for data manipulation, maintenance, and analysis
- OARdb database
  - Output of AFA captured in DB2 database as Availability Event and Job Interrupt tables
- Tools for managing OARdb records and calculating results
  - CLI and GUI for viewing, entering, editing the events
  - Calculate and store weekly MTTI, MTTF, Overall Availability, Scheduled Availability (replaced manual analysis)
  - Python based
- Added Weekly Root Cause Analysis
  - Weekly multi-hour meeting with Ops staff
  - Root cause analysis of all System and Unknown failures
  - Availability and Interrupt events annotated with results, re-categorized as User or System
  - Weekly OAR Master builds file of updated data to upload to OARdb

Mean Time To Interrupt Report Example

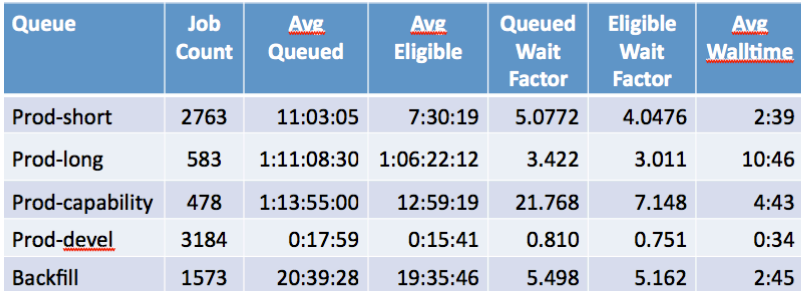
- Three report areas
  - Hardware only
  - All “System” failures
  - Component failure count
- Report headers
  - Resource
  - Type of records
- Column headers
  - MTTI: Mean Time to Interrupt, expressed in seconds and days
  - Events: number of interrupt events (job failures sharing same root cause)
  - Job Total: Total number of impacted jobs
  - Job Mean: Mean of jobs impacted per event



```
               ******************************************
               *  Mean Time to Interrupt (MTTI) Report  *
               ******************************************

Requested range: 2010-12-01 to 2011-03-04
   Generated on: Fri Mar  4 19:42:17 2011

Resource: Intrepid    Records: Hardware Only

 Period      MTTI s/days     Events  Job Total/Mean       Date Range
--------   --------------   ------  --------------   ----------------------
All         993600  11.50        8       46       5  2010-12-01/2011-03-03
Last 90     972000  11.25        8       46       5  2010-12-03/2011-03-03
Last 60     864000  10.00        6       14       2  2011-01-02/2011-03-03
Last 30    2592000  30.00        1        1       1  2011-02-01/2011-03-03

Trend, 28 day intervals:
Intvl  3   1209600  14.00        2       32      16  2010-12-10/2011-01-06
Intvl  2    604800   7.00        4       12       3  2011-01-07/2011-02-03
Intvl  1   2419200  28.00        1        1       1  2011-02-04/2011-03-03

Resource: Intrepid    Records: All Non-User Sources

 Period      MTTI s/days     Events  Job Total/Mean       Date Range
--------   --------------   ------  --------------   ----------------------
All         165600   1.92       48      200       4  2010-12-01/2011-03-03
Last 90     165446   1.91       47      198       4  2010-12-03/2011-03-03
Last 60     162000   1.88       32      101       3  2011-01-02/2011-03-03
Last 30     117818   1.36       22       47       2  2011-02-01/2011-03-03

Trend, 28 day intervals:
Intvl  3    186092   2.15       13       95       7  2010-12-10/2011-01-06
Intvl  2    268800   3.11        9       53       5  2011-01-07/2011-02-03
Intvl  1    109963   1.27       22       47       2  2011-02-04/2011-03-03

Component fault analysis for prior 60 days (2011-01-02/2011-03-03):

Component                 Count
--------------------      -----
scheduler                 13
machine                   6
gpfs                      6
myricom                   5
sn                        2
```

Impact of Improved Process and OARTool

- More accurate OAR results
  - Reflected consistent calculations and consistently applied business policy
  - Information on availability events gathered NLT 1 week from the event
  - Majority of Unknowns now characterized properly as User
- However, more interesting benefits began to emerge
  - Level of understanding of the very complex system increased across the Ops Team
  - Weekly immersion in job and system failures raised awareness and facilitated making connections between failures
  - Weekly summary of major component failures led to swat teams focused on underlying causes of system instability
- Regular root cause analysis implemented for scheduling as well
  - Increased understanding of scheduling complexities across whole facility
  - Modifications to reduce queue wait time
  - Able to track and see impact of changes



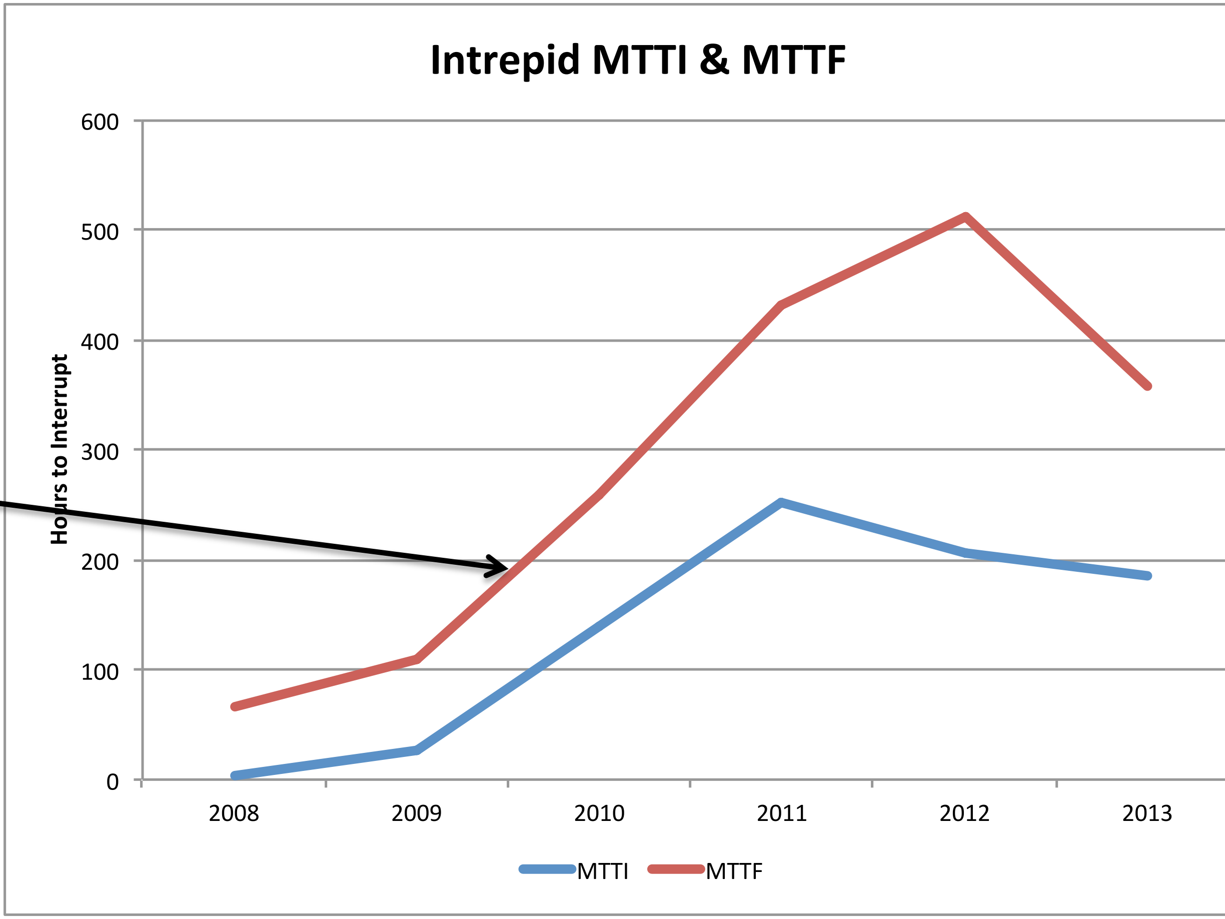
Queue	Job Count	Avg Queued	Avg Eligible	Queued Wait Factor	Eligible Wait Factor	Avg Walltime
Prod-short	2763	11:03:05	7:30:19	5.0772	4.0476	2:39
Prod-long	583	1:11:08:30	1:06:22:12	3.422	3.011	10:46
Prod-capability	478	1:13:55:00	12:59:19	21.768	7.148	4:43
Prod-devel	3184	0:17:59	0:15:41	0.810	0.751	0:34
Backfill	1573	20:39:28	19:35:46	5.498	5.162	2:45

Examples of Success Stories from the Process

- Large quantities of jobs failing due to boot failures
  - 9.1% of boots failing
  - Swat team deployed – purchased and deployed NAS, reconfigured central database
  - Boot failures went to 0, full machine boot went from 15 mins to 5 mins
  - 100x improvement in database performance and many other improvements
- Component fault report began showing GPFS as top contributor by large margin (16 out of 32 events)
  - Swat team deployed – network, gpfs, and service node cfg changes
  - GPFS dropped to a minor contributor (2 of 16 events)
  - System failure events cut in half, MTTF increased by 10%
- Large number of jobs failing due to failed I/O
  - Root cause analysis led to correlating the failures with another user's automated job submission script
  - Educated user, script fixed, I/O failures disappeared

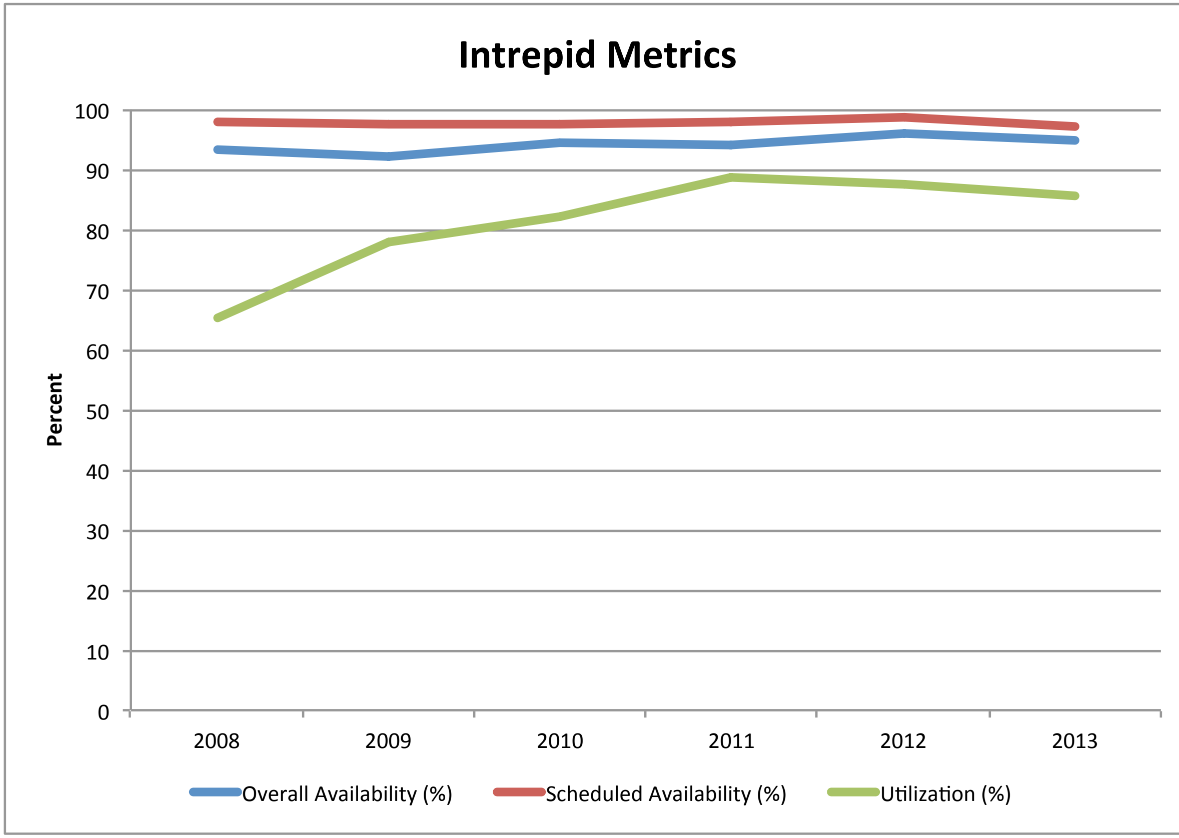
Intrepid MTTI and MTTF Over Lifetime

- MTTI is time to any outage
  - Failures
  - Scheduled outages
  - Max possible ~336 hrs
- MTTF is time to a system failure
  - Hardware & Software
- Root Cause Analysis implemented 2010
  - 2.5x improvement to MTTF
- Final year
  - Data Center plagued by power issues



Intrepid Availability & Utilization Over Lifetime

- Overall Availability
  - 92.1% in 2009 to 95.9% in 2012
- Scheduled Availability
  - 97.5% in 2009 to 98.5% in 2012
- Utilization
  - 78.1% in 2009 to
  - 87.6% in 2012
  - Anything over 80% is excellent
  - Attributed to root cause analysis for both job failures (and accompanying education of users) and scheduling



Weekly Root Cause Analysis Valuable but…

- Weekly Root Cause Analysis meeting were painful and time consuming
  - 4 hours or more each week
  - 5 or more people involved
  - JFA Master made all edits – not scalable
  - No view into what others were discovering during meeting
- New systems to be deployed in new data center
  - Next generation BG/Q system (Mira) with all new infrastructure
  - New RAS events with different meanings
  - Required porting of codes and tools
  - Took advantage to address biggest issues with the process
- Alacrify project to improve AFA and add QA and testing
- Storm project to improve Weekly Root Cause Analysis process
  - Front end for managing root cause analysis
  - Drag and drop jobs from one grouping to another
  - Multi-person editing and close to real-time viewing of changes
  - Tagging – text and colors from automated analysis of failures

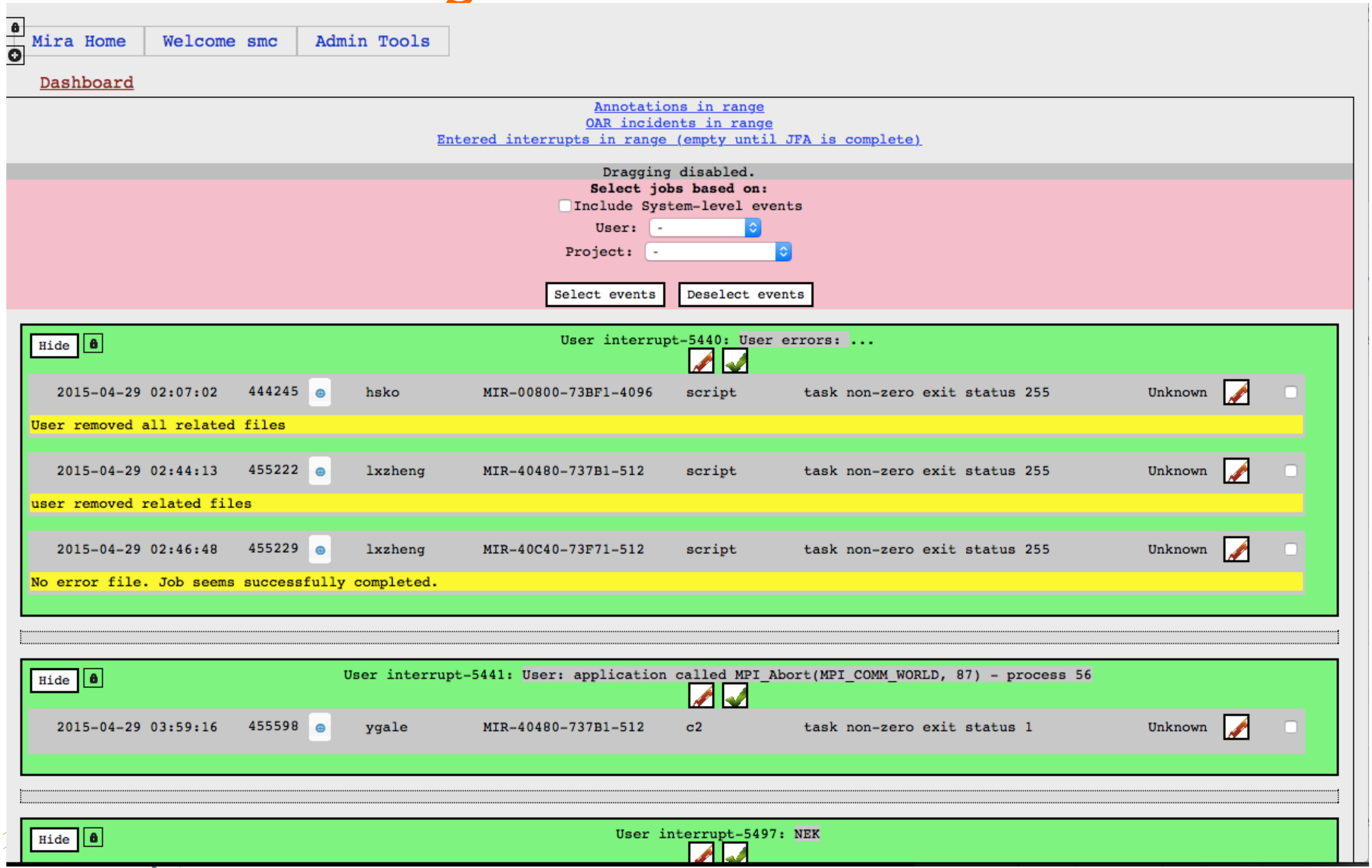
Alacrify Project

- Goal: Port to new systems and infrastructure, improve portability, add testing to Automated Failure Analysis code
- Rewrote and modularized AFA code
  - Converted to python libraries
- Added libraries with business logic for calculating metrics
- Improved QA
  - Heavily instrumented with unit tests
- Jenkins deployed to provide nightly testing
  - Unit tests for Alacrify libraries
  - Verification tests for availability events and job interrupts
  - Many others
  - Jenkins master has slave systems with special access to various restricted networks
- Implemented separate complete development and release environments
- Integrated with ALCF Data Warehouse

Storm Project

- Goal: Improve weekly Root Failure Analysis process
- Storm server – VM on a standard IT server
  - Apache
  - WSGI (Web Server Gateway Interface) application
  - django app (python)
- Storm provides weekly Job Failure Analysis (JFA) interface
  - Java script doing AJAX calls
  - Script accesses django app and requests data
  - Uses RabbitMQ to manage message queues
- Close to real-time updates during JFA (every 10s)
  - Ops staff log into JFA page
  - Individual RabbitMQ message queues auto-generated on login
  - When staff makes a change, the webserver writes to the db and sends rmq messages to all message queues
  - In separate threads for each person, Ajax polls their queue to see if they have a message, then takes the message and calls a java script to update the screen, removing the message from the rmq
- Many cool features including job, power, temp real-time graphs
- Reduced weekly meeting time to around an hour instead of over 4

Storm JFA Page



Mira Home | Welcome smc | Admin Tools

Dashboard

Annotations in range
OAR incidents in range
Entered interrupts in range (empty until JFA is complete)

Dragging disabled.

Select jobs based on:

□ Include System-level events

User: -

Project: -

Select events | Deselect events

Hide

User interrupt-5440: User errors: ...

2015-04-29 02:07:02	444245	hsko	MIR-00800-73BF1-4096	script	task non-zero exit status 255	Unknown	□
User removed all related files							
2015-04-29 02:44:13	455222	lxzheng	MIR-40480-737B1-512	script	task non-zero exit status 255	Unknown	□
user removed related files							
2015-04-29 02:46:48	455229	lxzheng	MIR-40C40-73F71-512	script	task non-zero exit status 255	Unknown	□
No error file. Job seems successfully completed.							

Hide

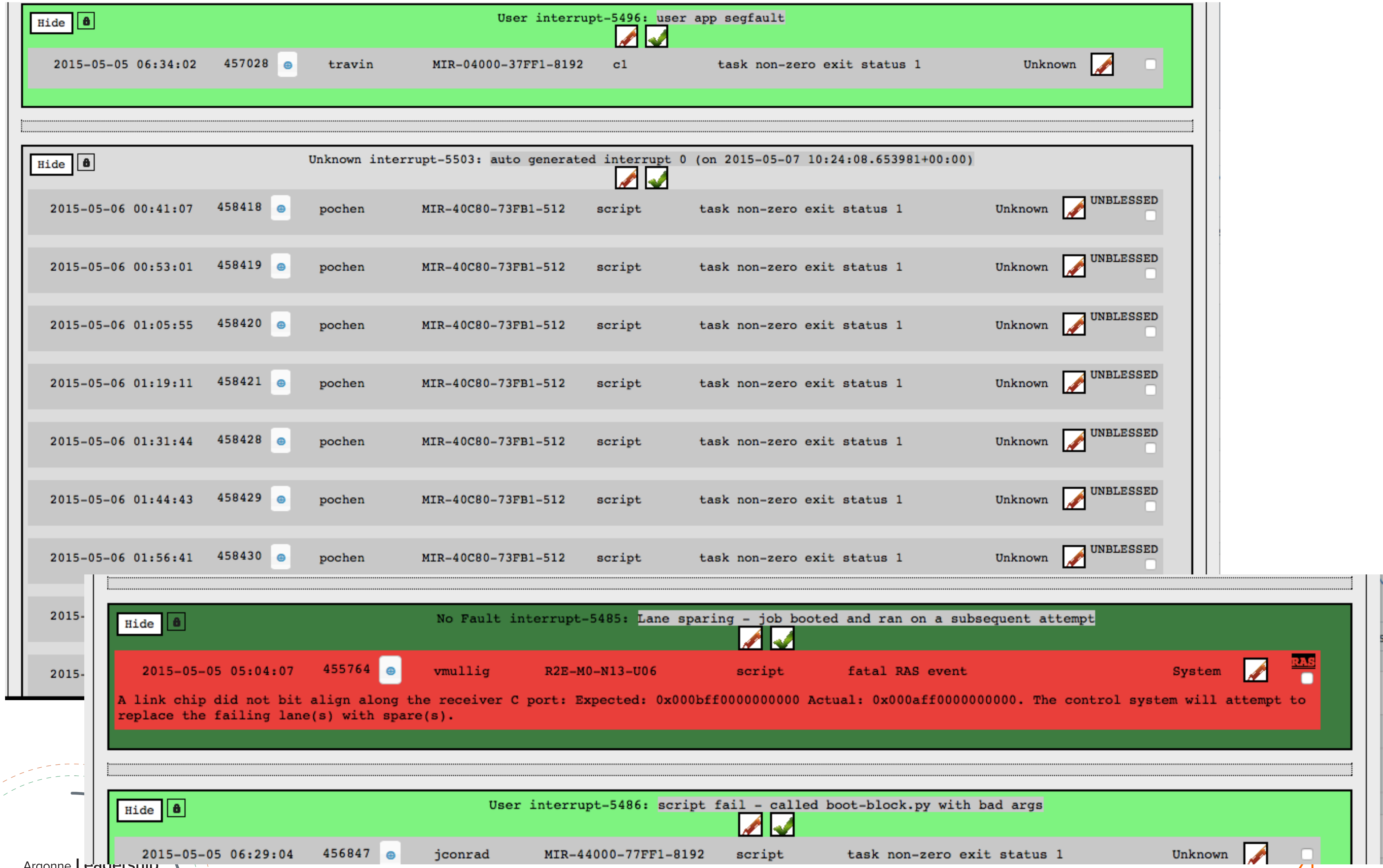
User interrupt-5441: User: application called MPI_Abort(MPI_COMM_WORLD, 87) - process 56

2015-04-29 03:59:16	455598	ygale	MIR-40480-737B1-512	c2	task non-zero exit status 1	Unknown	□

Hide

User interrupt-5497: NEK

Storm JFA Examples



Hide 🔒 User interrupt-5496: user app segfault

```
2015-05-05 06:34:02   457028   travin   MIR-04000-37FF1-8192   c1   task non-zero exit status 1   Unknown   □
```

Hide 🔒 Unknown interrupt-5503: auto generated interrupt 0 (on 2015-05-07 10:24:08.653981+00:00)

```
2015-05-06 00:41:07   458418   pochen   MIR-40C80-73FB1-512   script   task non-zero exit status 1   Unknown   UNBLESSED □
2015-05-06 00:53:01   458419   pochen   MIR-40C80-73FB1-512   script   task non-zero exit status 1   Unknown   UNBLESSED □
2015-05-06 01:05:55   458420   pochen   MIR-40C80-73FB1-512   script   task non-zero exit status 1   Unknown   UNBLESSED □
2015-05-06 01:19:11   458421   pochen   MIR-40C80-73FB1-512   script   task non-zero exit status 1   Unknown   UNBLESSED □
2015-05-06 01:31:44   458428   pochen   MIR-40C80-73FB1-512   script   task non-zero exit status 1   Unknown   UNBLESSED □
2015-05-06 01:44:43   458429   pochen   MIR-40C80-73FB1-512   script   task non-zero exit status 1   Unknown   UNBLESSED □
2015-05-06 01:56:41   458430   pochen   MIR-40C80-73FB1-512   script   task non-zero exit status 1   Unknown   UNBLESSED □
2015-
2015-
```

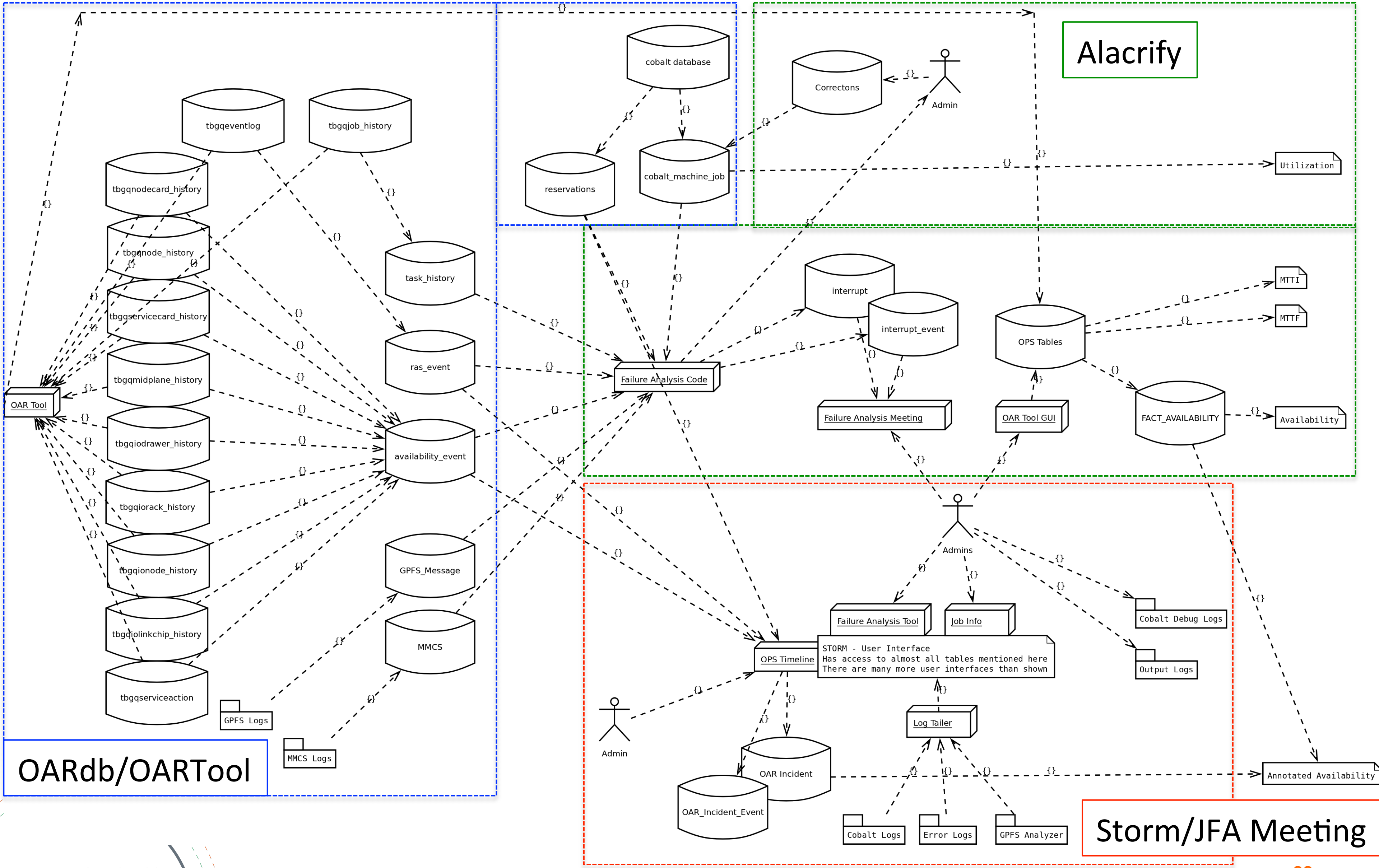
Hide 🔒 No Fault interrupt-5485: Lane sparing - job booted and ran on a subsequent attempt

```
2015-05-05 05:04:07   455764   vmullig   R2E-M0-N13-U06   script   fatal RAS event   System   RAS □
A link chip did not bit align along the receiver C port: Expected: 0x000bff0000000000 Actual: 0x000aff0000000000. The control system will attempt to
replace the failing lane(s) with spare(s).
```

Hide 🔒 User interrupt-5486: script fail - called boot-block.py with bad args

```
2015-05-05 06:29:04   456847   jconrad   MIR-44000-77FF1-8192   script   task non-zero exit status 1   Unknown   □
```

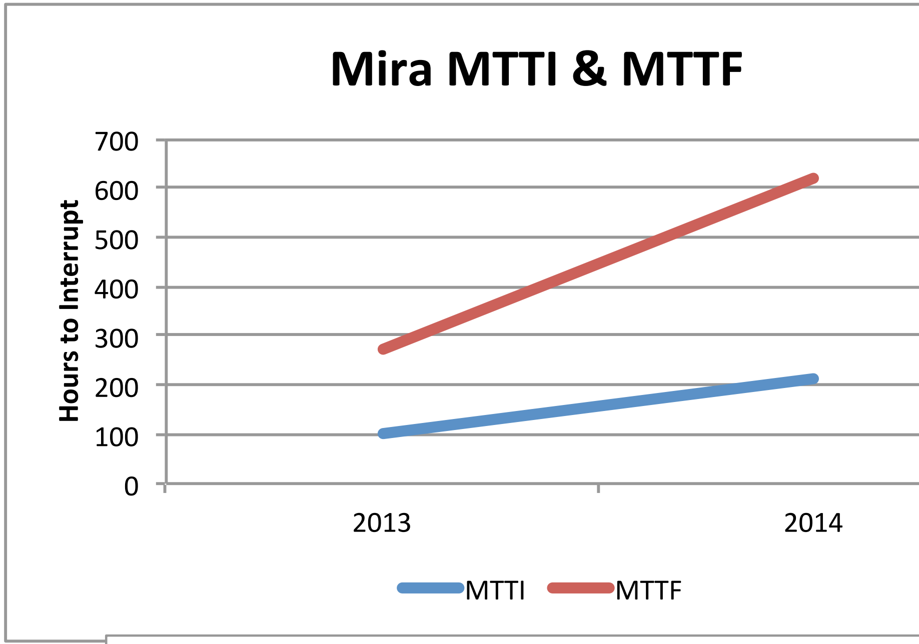
Current Process

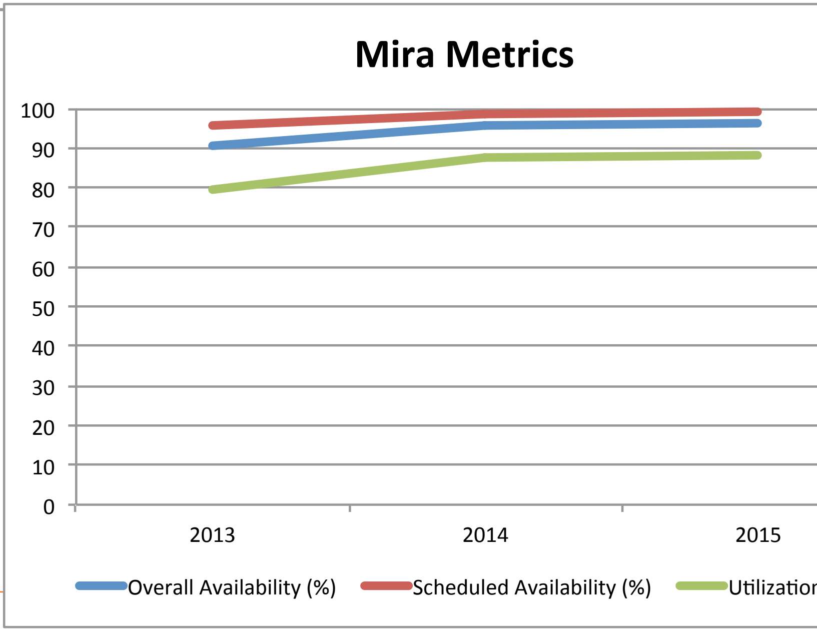


STORM - User Interface
Has access to almost all tables mentioned here
There are many more user interfaces than shown

Impact of this process and tools on Mira

- Mira is a 20% larger system
  - By number of nodes
  - Even larger by component counts
- Even with that, it is very stable
- MTTI/MTTF started where Intrepid was in year 2
  - 2nd year of Mira exceeds Intrepid's best MTTF by 20%
- Overall and Scheduled Availability are already over Intrepid's best
  - 96.4% and 99.4%
  - Even with multiple power outages
- Utilization started at 79.4% and is now at 88.1%





Future Work

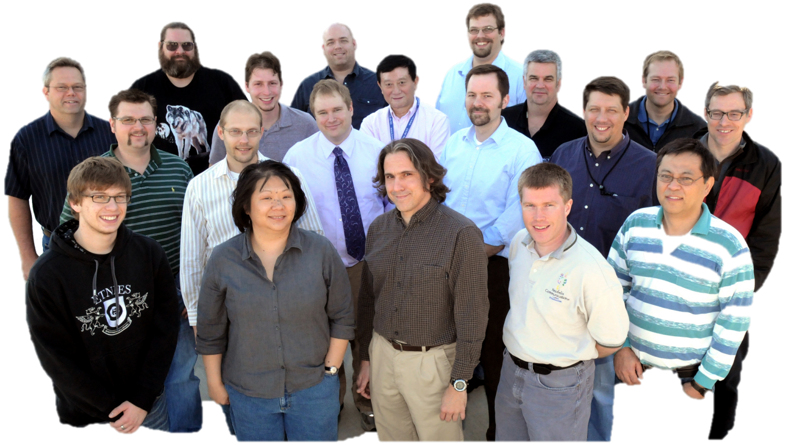
- Porting of tools/codes/process to non-Blue Gene systems
  - Current tools are Blue Gene centric, not really usable by non-BG sites
  - Two new ALCF systems just announced for 2016 and 2018
    - Theta – Intel/Cray XC-40 with 2nd Gen Intel Xeon Phi (Knights Landing – KNL)
    - Aurora – Intel/Cray Shasta with 3rd Gen Intel Xeon Phi (Knights Hill – KNH)
  - Porting of OAR and FA tools, codes, and process will be required
  - Rework to remove Blue Gene-isms - potential for public release
- Improve automated failure analysis to add additional correlation capability, incorporate additional data sources
  - Vast majority of failures categorized as Unknown are really User
- Add capability to easily bring User jobs back into root cause analysis process
  - Ability to search and automatically pull in failures records incorrectly categorized as User
- Potentially replace WSGI with Websockets
  - Would greatly simplify Storm

Summary

- Original driver for weekly root cause analysis was to meet DOE requirements for reporting metrics
  - OAR results are now accurate, consistently generated, and retained in a database
- True value lay in deep and wide root cause analysis of every job failure and every availability event
  - Data gathered on real cause of failures over time
  - Focused team on underlying causes of system instability
  - Used to drive improvement and upgrade planning
  - Contributed to improved MTTI/MTTF, Availability, and Utilization
  - Insight into users behaviors used to educate users and improve schedule
  - Increased Ops knowledge and expertise of complex systems
- Direct contributor to stabilizing Mira so quickly

Credits

- Work presented today was developed over the past 7 years by a lot of people
- Automated Failure Analysis Project
  - Primaries: Brian Toonen and Andrew Cherry
- OARdb and OARTool Project
  - Primaries: Cheetah Goletz and Brian Toonen
- Storm Project
  - Primaries: Eric Pershey and Nick Anderson
- Alacrify Project
  - Primaries: Nick Anderson and Eric Pershey
- Along with everyone who has worked on the ALCF Operations Team



Thank you