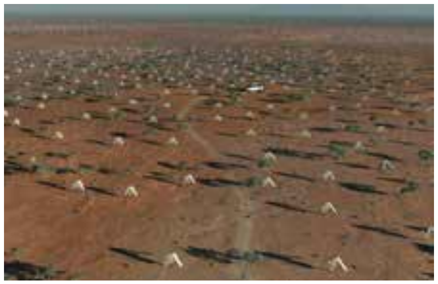
10^{18} BDEC

Exec Committee
Pete Beckman
Jean-Yves Berthou
Jack Dongarra
Yutaka Ishikawa
Satoshi Matsuoka
Philippe Ricoux

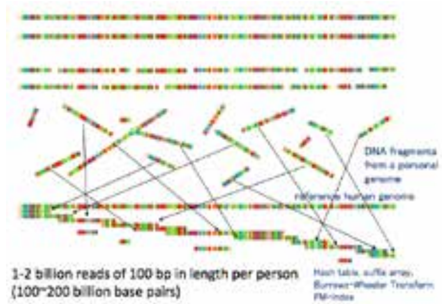
BIG DATA AND EXTREME-SCALE COMPUTING

- Overview of Previous Meeting
- Summary of Key Concepts & Strawman
- Plan for Breakouts
  - Why you are here….

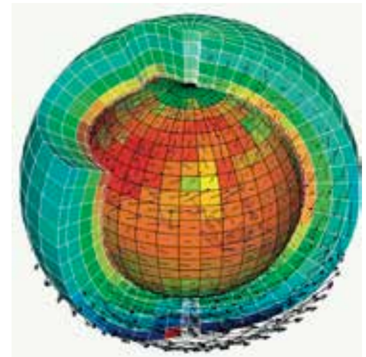
Application Drivers & Use Cases



SKA: Tom Cornwell

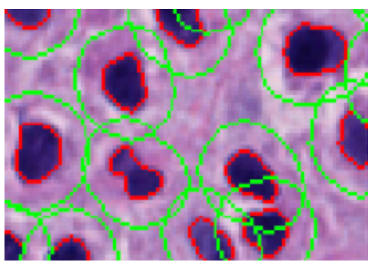


Genomics:
Shinichi Morishita

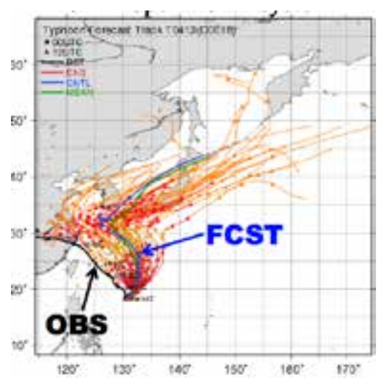


Climate: Pier Luigi Vidale,
Malcolm Roberts

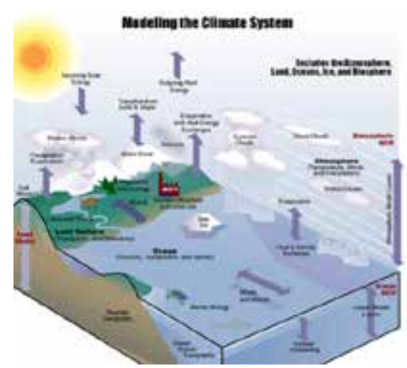
Cosmology:
Jean-Michel Alimi



Medical: Joel Saltz



Weather: Takemasa Miyoshi



Climate: Sandro Fiore

etc, etc, …

Architecture/Software

BD Usage Models Differ from EC

Big Data
- Continuous access require based on data generation/ submission rates
- CPU time, I/O and data volume all important
- Data products typically used in future computations via an integration or pipeline
- Data products made available for external users and curated over time

Extreme Compute
- Batch oriented access based on allocations for specific projects
- Mostly CPU time centric
- Output not necessarily used in future runs but often significant time used for visualization
- Output generally (but not always) used "privately" and rarely curated

Rick Stevens

BLUE WATERS

Resource Management

- Resource Scheduling Functions on large scale systems have the features needed to schedule work as people expect for Big Data
  - Scheduling decisions are based on "culture" that is made up of users, providers, stack holders, etc.
  - Queuing theory determines the tradeoffs of utilization vs response time
  - In "Big Data" "batch" background work is done for what we perceive as interactive query response
    - E.g. crawling and indexing web pages, image preparation, weather products, ...
- Changes required
  - Need to schedule bandwidth – not processors
  - How many units need to be schedule as a unit
  - For Big Data – need to move computation to the data or move the data
  - Can do at least space sharing within a system

William Kramer

Exascale Hardware/Software Architecture

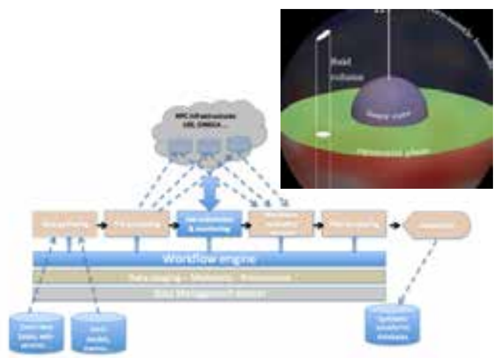
- *Need to stage very large datasets for relatively short periods of time* -- large aggregate bandwidth to non volatile scratch storage -- distributed flash and disk
- *Globally addressed/indexed persistent data collections* -- e.g. DataSpaces, Region Templates (GIS analogy), persistent PGAS
- *Intelligent I/O with in-transit processing*, data reduction (e.g. ADIOS)
- Visualizations need to be carried out interactively and in situ as data is produced and as computations proceed – efficient streaming data

Joel Saltz

Architectural Challenges

- How to build a system for the posterior analysis?
- Where should data be stored
  - *Not directly at the supercomputer (too expensive storage)*
  - *Computations and visualizations must be on top of the data*
  - *Need high bandwidth to data source*
- Scheduling of complex I/O access patterns
  - *Databases are a good model, but are they scalable?*
  - *Google (Dremel, Tenzing, Spanner: exascale SQL)*
  - *Augmented with value-added analytic services (SciDB, etc)*
- Data organization
  - *Cosmology simulations are not hard to partition (scale-out)*
  - *Use fast, cheap storage for data streaming (sequential)*
  - *Consider a tier of large memory systems (random access)*

Alex Szalay



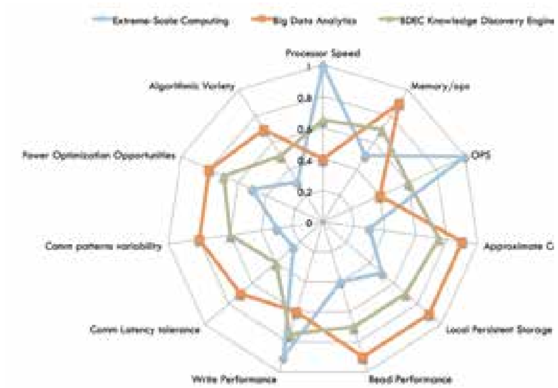
Jean-Pierre Vilotte

Some Architecture Issues for Big Data

- Parallelism in I/O
  - Systems optimized for zillion independent files or records can use cloud resources
  - Deeper hierarchy in I/O system
    - BW example: 26 PB disk, 380 PB tape with 1.2 PB cache for the 26 PB cache; use of RAIT to improve performance, reliability
    - Important distinction for extreme scale systems: All data accessible at nearly same performance from *all* nodes
  - Metadata design has a major impact on performance, reliability
- Other architectural features important
  - One-sided access with remote operations
    - At least multi-element compare-and-swap
    - Even better, compute to data (active messages, parcels, ...)
  - And others (better stream processing, custom control logic...)

10 PARALLEL@ILLINOIS

William Gropp



Alok Choudhary

Data Analysis

- Two fundamental aspects
  - Pattern matching: Perform analysis tasks for finding known or expected patterns
  - Pattern discovery: Iterative exploratory analysis processes of looking for unknown patterns or features in the data
- Ideas for the analysis of Big Data
  - Perform pattern matching tasks in the simulation machine
    - "In situ" analysis
  - Prepare data for pattern discovery on the simulation machine, and perform analysis on mid-size analysis machine
    - "In-transit" data preparation
    - "Off-line" data analysis

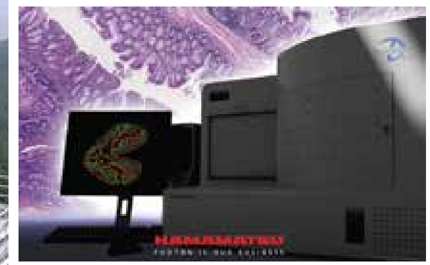
Arie Shoshani

Instruments & Facilities

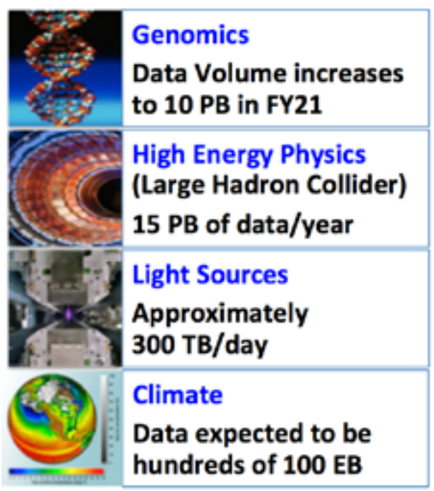
- “HPC Instrument” (Tsubame, Mira)
- SDSS, LSST, SKA, LOFAR , …
- APS(20x), SNS, …
- DNA Sequencers
- LHC / Atlas
- ARM

HPC

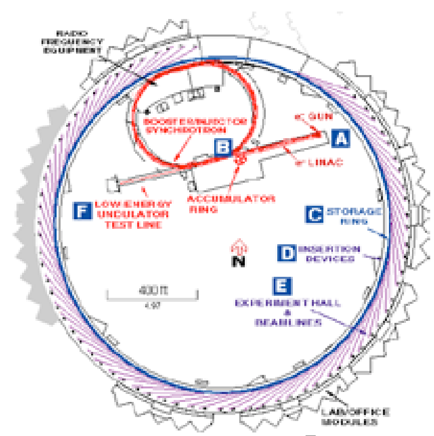
Sloan DSS



Tissue Samples

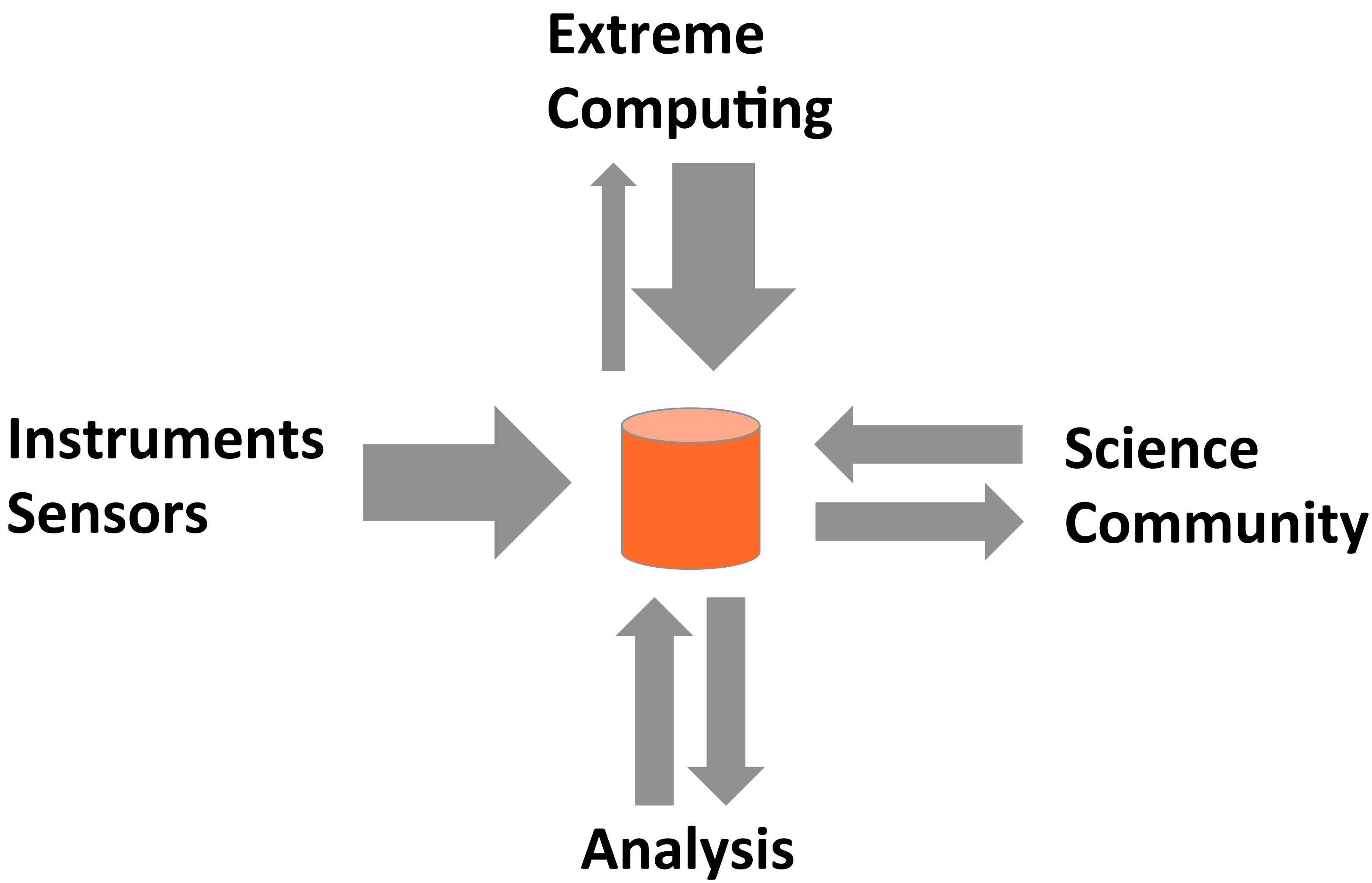


ARM

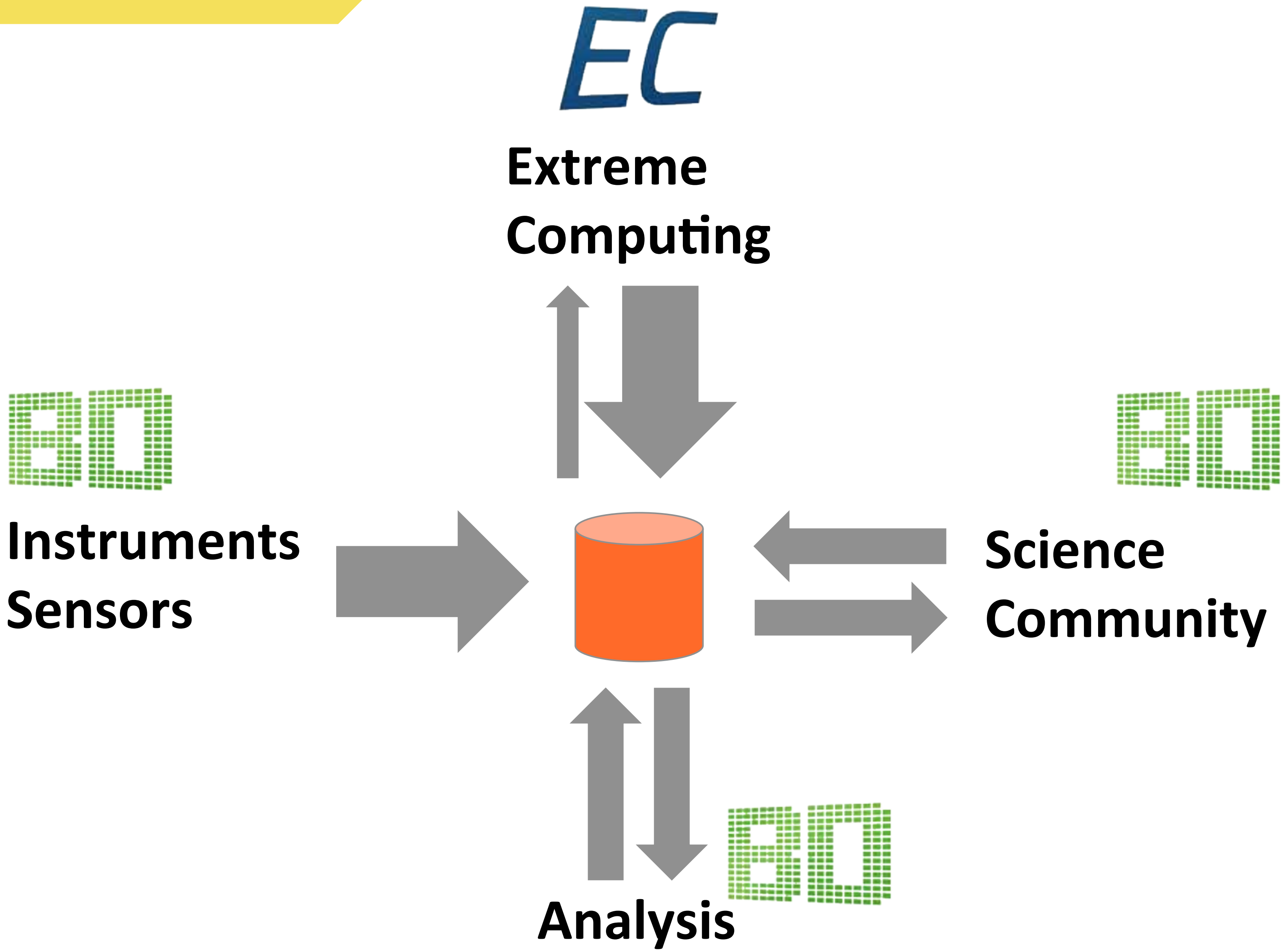


APS

Data-Centric View



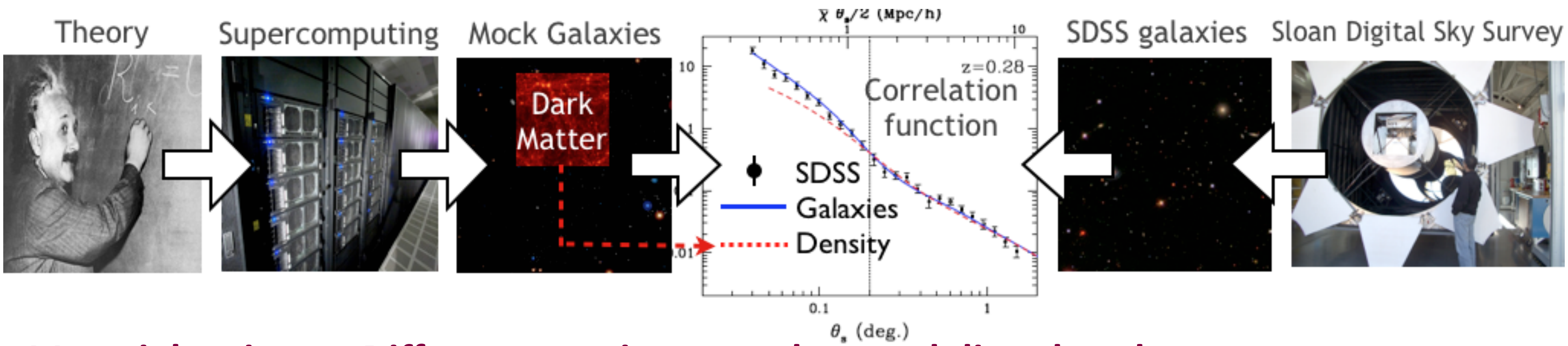
Data-Centric View



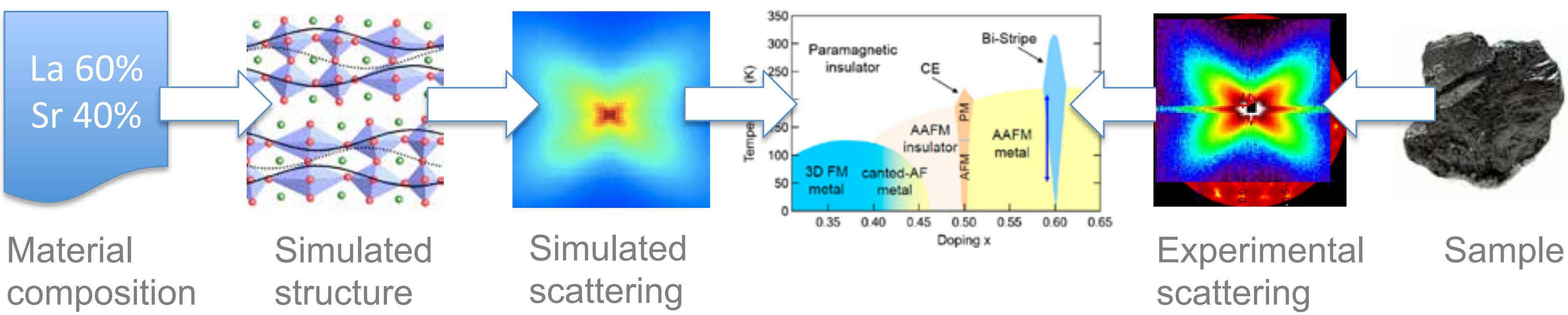
Examples of coupling:

observation (measurement) and computation (simulation)

Cosmology: The study of the universe as a dynamical system



Materials science: Diffuse scattering to understand disordered structures



Images from Salman Habib et al. (HEP, MCS, etc.) and Ray Osborne et al. (MSD, APS, etc.)

Outline: Summary and Straw man

1. Scientific Big Data Computing is Different
2. Strawman Architecture
3. Implementation Issues
4. Programming Models
5. Research Gaps & Needs
6. What Would a Facility Look Like?
7. BDEC Breakouts

Comparing Architecture

Big Data	Extreme Computing
? Cost in memory and interconnect bandwidth	***Significant Cost*** in memory and interconnect bandwidth
Little Cost for resilient hardware in data storage	***Significant Cost*** in resilient hardware in shared file system
Little Cost for hardware to support system-wide resilience	***Significant Cost*** in resilience hardware to reduce whole-system MTTI
Significant Cost: ***increased aggregate IOPs***	Significant Cost: ***cutting-edge CPU performance features***
Often trades performance for capacity	***Often trades capacity for performance***

Comparing Operations

Big Data	Extreme Computing
Continuous access to long-lived "services" created by science community	***Periodic access*** to compute resources via job submitted to scheduler and queue
Time-shared access to elastic resources	***Space-shared*** compute resources for exclusive access during jobs
New hardware capacity ***purchased incrementally***	New tightly integrated system ***purchased every 4 years***
Users charged for all resources (storage, cpu, networking)	***Users charged for CPU hours, storage and networking is free***

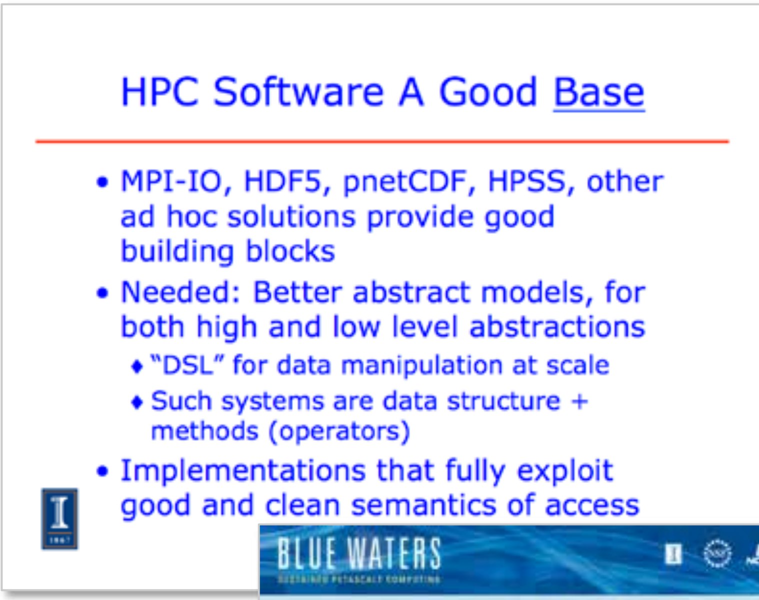
Comparing Software

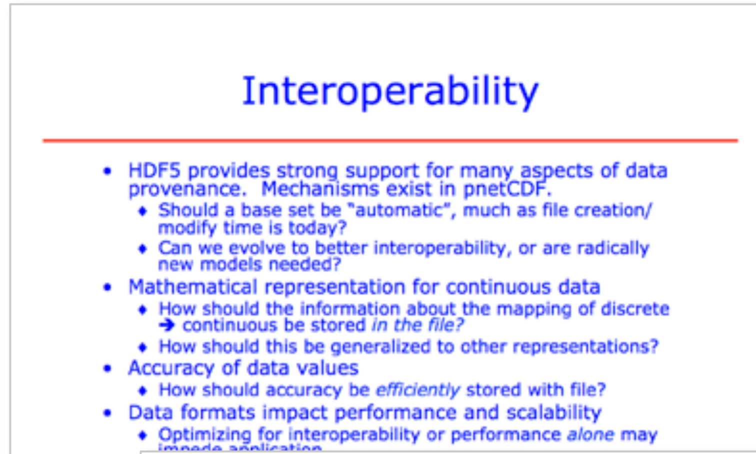
Big Data	Extreme Computing
Software responds to elastic resource demands	After allocation, ***resources static until termination***
Data access often ***fine-grained***	Data access is ***large bulk*** (aggregated) requests
Services are resilient to fault	***Applications restart after fault***
Often ***customized*** programming models	Widely ***standardized*** programming models
Libraries help ***move computation to storage***	Libraries help ***move data to CPUs***
Users routinely deploy their own services	***Users almost never deploy customized services***

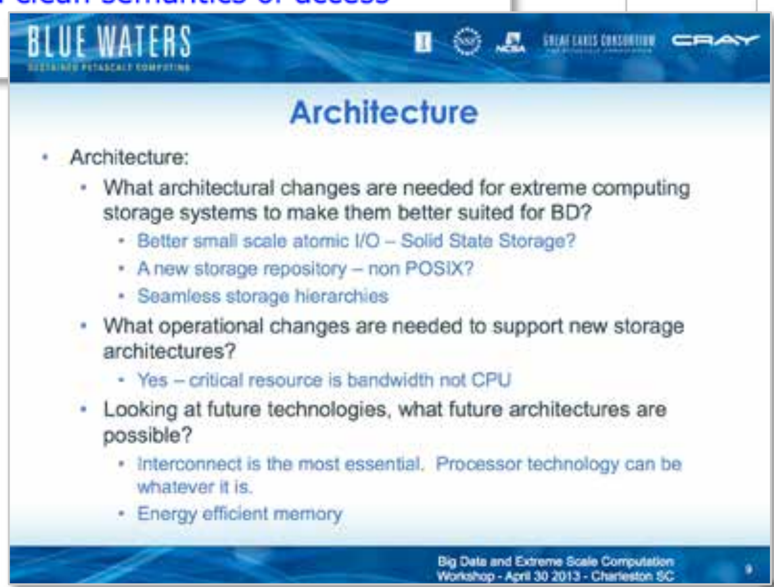
Comparing Data

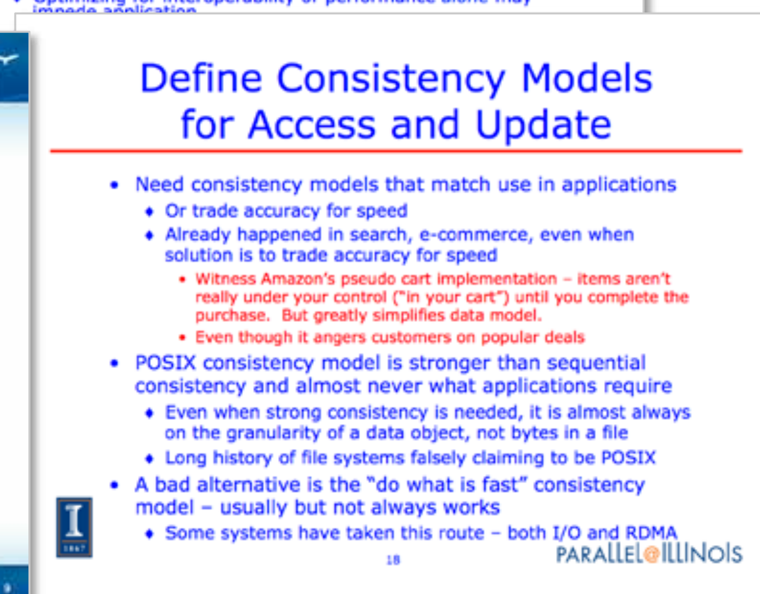
Scientific Big Data	Extreme Computing
Inputs ***arrive continuously***, streaming workflows	Inputs ***arrive infrequently***, buffering carefully managed
Data is ***unrepeatable*** snapshot in time	Data often ***reproducible*** (repeat simulation)
Data generated by sensors (***error: from measurement***)	Data generated from simulation (***error: from simulation***)
Data rate ***limited by sensors***	Data rate ***limited by platform***
Data often ***shared and curated*** by community	Data ***often private***
Often ***unstructured***	***Semi-structured***

What can we apply from EC to BD?









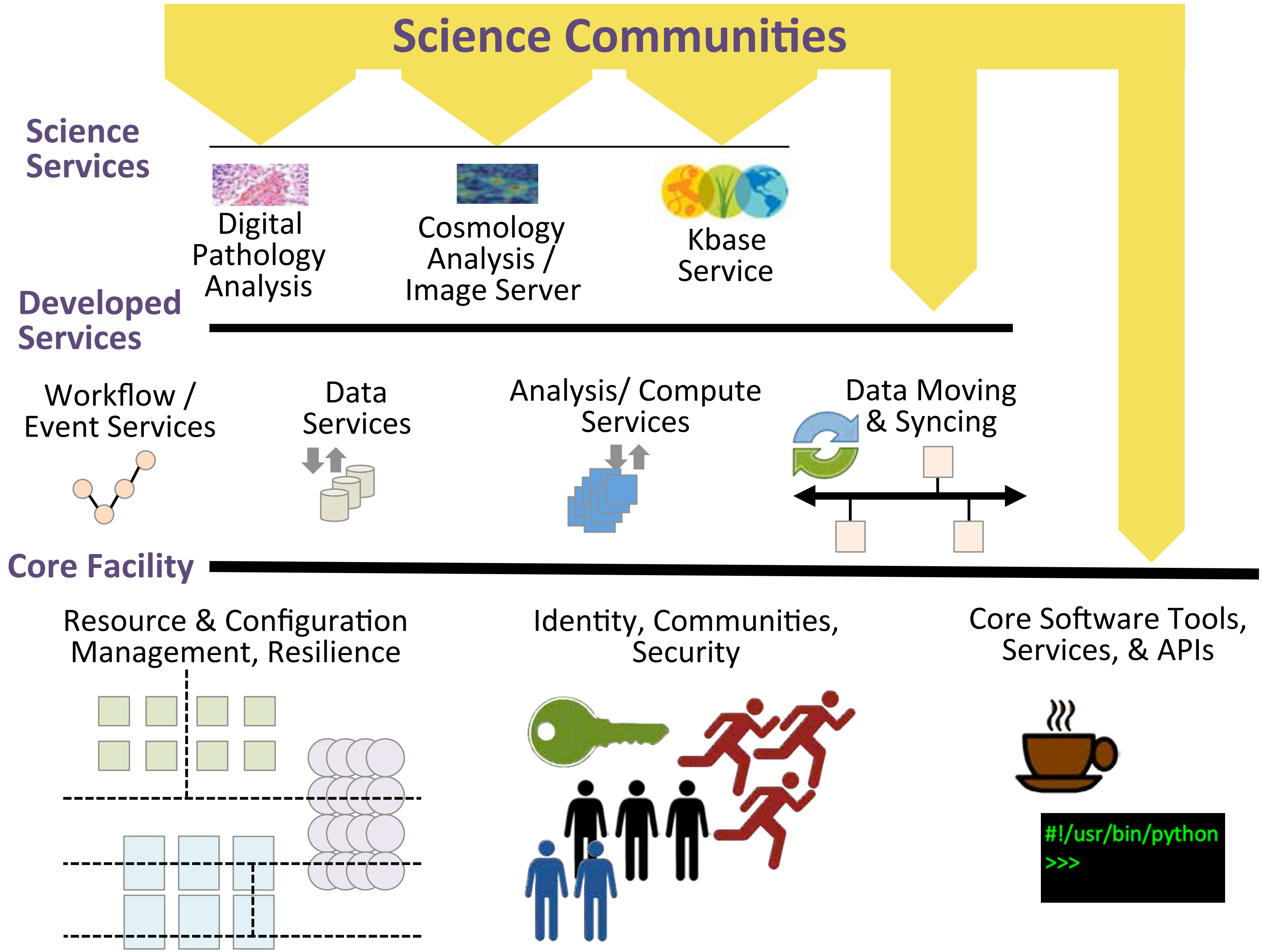
William Gropp
UIUC
William Kramer
NCSA

What can we apply from BD to EC?

- Hmmm, very very good question…
  - Not as much exploration of this yet
  - Changing operational & cost models
  - Supporting persistent services
  - Virtualization to address software complexity?
  - …

Take Away Messages

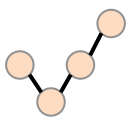
- EC-classic is morphing into BDEC
  - The "info-plosion" makes this inevitable…
- Paradigms and abstractions similar
- Lessons learned from EC can often be applied
- BD ***software tools/layers are significantly more diverse*** than EC-classic
  - Often this fuels the cloud IaaS discussion


Science Communities
Science Services
Digital Pathology Analysis
Cosmology Analysis / Image Server
Kbase Service
Developed Services
Workflow / Event Services
Data Services
Analysis/ Compute Services
Data Moving & Syncing
Core Facility
Resource & Configuration Management, Resilience
Identity, Communities, Security
Core Software Tools, Services, & APIs
#!/usr/bin/python
>>>

Technology Bits: Many Examples

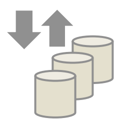
Developed Services

Workflow / Event Services



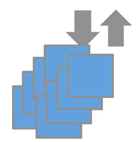
Taverna
Kepler
VisTrails
DAGMan
Pegasus
Chiron
Swift
…

Data Services



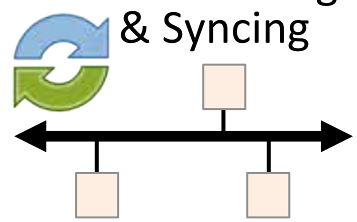
SciDB
S3
Cassandra
HDFS
EBS
MemCache
SQL/DB2/Oracle
Key/Value
…

Analysis/ Compute Services



MapR
Paraview
VisIt
Pregel
R
Pegasus
ScaleGraph
….

Data Moving & Syncing

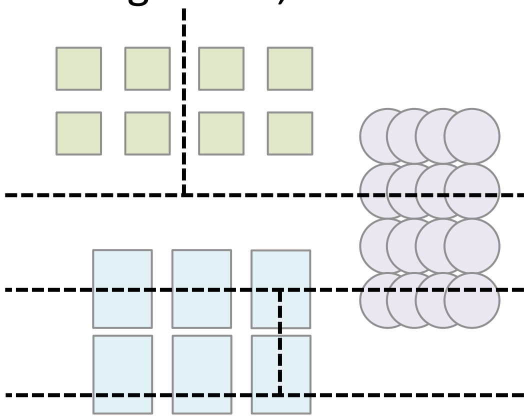


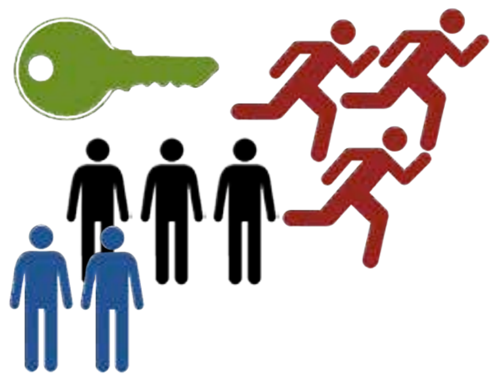
Globus Online
CATCH
Data Pipelines
…

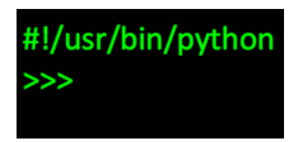
Technology Bits: Many Examples

Core Facility

Resource & Configuration Management, Resilience	Identity, Communities, Security	Core Software Tools, Services, & APIs







Resource & Configuration Management, Resilience:

OpenStack
EC2
LXC
Omega
OpenVZ
VMWare
Apache Mesos
Omega
…

Identity, Communities, Security:

GSSAPI
Shibboleth
InCommon
…

Core Software Tools, Services, & APIs:

…

Impact of Programming Model
(more work needed during breakout sessions)

- Workflow
  - Data movement, Events, pub/sub
- Composition
  - <n> parallel programs coupled and sharing data
- Elasticity
  - Interfaces for give/take, predict/reserve
- Co-location
  - Data & Compute

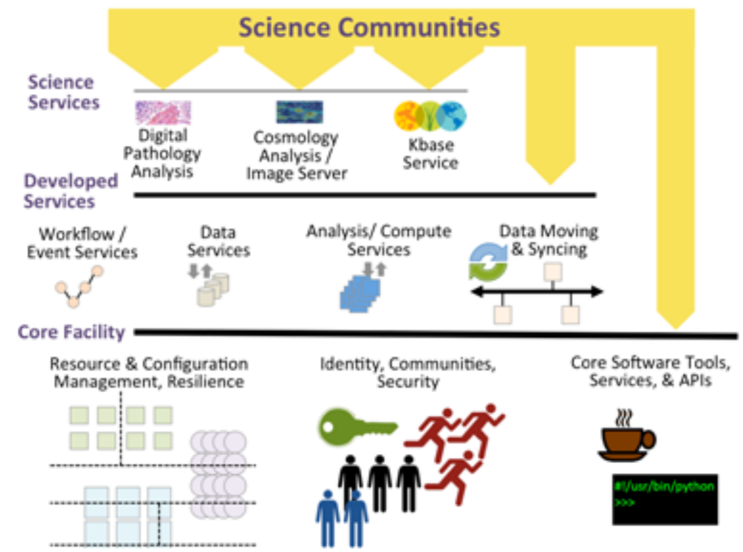
Current Gaps & Needs

- This meeting should help ***frame*** BDEC
  - Don't solve the problem, or get mired in tech

- Work toward ***identifying the research questions and promising directions***, not the answers, or how to spend other people's money

Extending Current HPC Facilities for BDEC
A new kind of facility?

- **The model is different from an HPC center**
- **Is the "programming environment" replaced with workflows of services?**



- Science Domains make long term commitments to facility
- Facility Staff:
  - Domain Science CS: Develop specialized capabilities
  - 50/50: Design and develop domain services
  - BDEC SysAdmin: Develop and support core services

Breakout: Applications

- Session 1: 90 Minutes
  - (quickly) Describe 6 to 8 BDEC workflows (bio, cosmology, climate, etc). Cover different types (stream processing, extract/subset, analysis, etc). Include:
    - Current best practice for science community
    - Integration with HPC community?
    - Ideal future design?
    - Describe and classify workflow steps (specific, with as must detail as time permits)
  - Report out during Plenary
- Session 2: 120 Minutes
  - Analyze and discuss the initial work of the other breakouts
  - Describe:
    - Application perspective: Common/Basic services for a BigData system,
    - Operational models (sharing resources, scheduling, identity management) and tradeoffs in design
    - Mini-apps that can be constructed for the workflows
  - Report out during Plenary

Breakout: Architecture

- Session 1: 90 Minutes
  - Describe a straw man architecture with Common/Basic services for BDEC system. Include:
    - Core services (e.g. cloud? Database? Identity management?)
    - Integration with HPC community?
    - Science-optimized services (e.g. parallel storage, data syncing and mv, sharing, etc).
  - Report out during Plenary
- Session 2: 120 Minutes
  - Analyze and discuss the initial work of the other breakouts
  - Describe:
    - How App workflows can be supported by Common/Basic services
    - Operational models (sharing resources, scheduling, identity management) and tradeoffs in design
    - How well does straw man architecture meet application and data needs?
    - Core benchmarks / measurements for architecture of BDEC system
  - Report out during Plenary

Breakout: Data

- Session 1: 90 Minutes
  - Describe what is needed in Common/Basic data services for BDEC system
    - What are the most accepted (deployed) services?
    - What are the largest gaps for BDEC science communities?
    - What kinds of optimizations / specializations are required for science communities?
  - Report out during Plenary
- Session 2: 120 Minutes
  - Analyze and discuss the initial work of the other breakouts
  - Describe:
    - What is needed to support Federation, Provenance, and Curation?
    - What are the programming models needed?
    - What basic services should be included in BDEC facilities first?
  - Report out during Plenary