3DPAS February Meeting

Wiki: http://wiki.esi.ac.uk/3DPAS-Working-2
White Paper:
https://docs.google.com/viewer?a=v&pid=explorer&chrome=true&srcid=0B9nxKNHtLPy2YTE0ODM4ZjMtNTcxMS00MTFmLTgwMGMtN2IzNzc5NjVmMTk2&hl=en

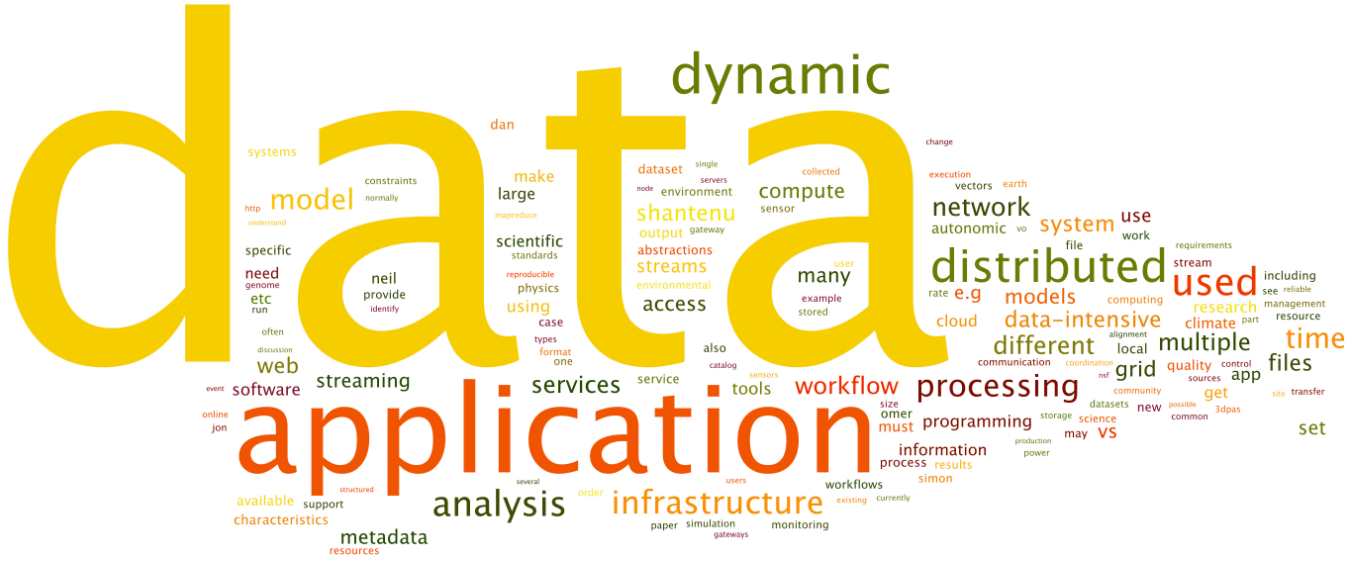
Participants:
Shantenu Jha (LSU)
Yanif Ahmad (Johns Hopkins)
Steve Fisher (Rutherford Appleton Lab / STFC)
Andre Luckow (LSU)
Don Middleton (NCAR)
Daniel Katz (Chicago)
Ole Weidner (LSU/Edinburgh)
Neil Chue Hong (EPCC)
Adam Carter (EPCC)
Yike Guo (Imperial)
Yin Chen (Edinburgh)
Jennifer Schopf (Woods Hole/NSF)
Adam Barker (St. Andrews)
Simon Dobson (St. Andrews)
Roy Sterritt (Ulster)
Jon Weissman (UMN)
Omer Rana (Cardiff Univ.)

Summary:

This workshop brings together established major contributors from different data-intensive application domains as well the computer science domains (parallel computing, Grid/Cloud as well data-intensive programming models and abstractions, distributed systems). The specific topics for this meeting are:

- What should this theme try to address? Must address? Must not address?
- What kind of applications should this theme focus on?
    - Climate and Earth Modeling
    - WikiAnalytics
    - ...
- What are the common applications and application characteristics?
- How to characterise the 3D data (dynamic, data-intensive, distributed)? What additional characteristics exists (diversification, reproducible, policies, archiving,...)?
- How do existing infrastructure handle 3D data?
    - Earth System Grid
    - DBToaster

    - Production Data
    - ELVIRA
    - ...
- What are the common underlying infrastructure issues?
    - Metadata management
    - Programming abstractions
    - Querying vs. programming vs. reasoning
- What role play autonomic approaches?



Day 1
Introduction Shantenu

Don Middleton: Cyberinfrastructure and the Global Environmental Data Challenge

3DPAS: data-intensive, distributed, dynamic, diverse + complex, damn big
Models: the community climate system model (CCSM), http://www.cesm.ucar.edu/models/
Multi-model analyses
Earth system data integration challenge:

- massive data archives (XBs)
- multiple data centers
- existing IT infrastructures
- heterogeneous data sources
- multiple physical realms
- multiple data, metadata formats
- multiple scales
- multiple audiences

- cybersecurity

Earth System Grid (ESG)

- provide access to all data that we produce
- collaborative distributed development
- http://www.earthsystemgrid.org
- exponential increase in data volumes:
    - 20000 registered users
    - >250 TB of climate simulation data available
    - 850 GB/day data delivered
    - model source codes, initialization datasets
- in Feb 2011: 1 PB data delivered

Related projects:

- Science Gateway framework (SGF) in development
- NARCCAP, Cooperative Arctic Data and Information Service (CADIS), NCAR's community data portal, TG, …
- TIGGE, http://tigge.ucar.edu/
- Digital Preservation

Major drivers for ESG Development

- 40+ models
- 10 PB distributed data produced
- replication of archive to several other sites
- most of data needs to be published by end 2011
- current round of IPCC (CMIP5) must be finished by 2013

Earth System Grid Federation (ESGF) is a distributed data archival and retrieval system
Gateways for ESG:

- Gateways (FTP, Security)
- Federation between Gateways using federation protocol (for data & metadata)
- Protocols: SAML, OpenID
- SSO
- Automated GUI-based publication tools
- data aggregation
- notification services
- Technology: Java, Spring, Tomcat
    - some scalability issues - some rearchitecting necessary
- Data Nodes:
    - services for analysis clients/dml
- data replication: local copy of data for scientists, reduce latencies, fault tolerance
    - data movement
        - data movement agent

        - globus-url-copy
        - Globus Online
    - dataset scanned and new THREDDS catalog generated
- Neil: What's the rate of change of these datasets?
    - At first there's a lot of thrashing
    - After a year or so has gone by, things are pretty static
    - but need to make sure data that is retracted/fixed is retracted/fixed everywhere
    - Neil: Not dissimilar to what's been seen with QCD data in the UK
- Earth System Curator: bridging models, analysis and data
- METAFOR, http://metaforclimate.eu
- Every computational scientist has another definition of grid => work on GridSpec
- ESG Dataflow consists of several quality checks
- Analysis/Visualization
    - Portal-based visualization (eventually must work on PB of data)
    - NCAR command language (NCL) (single threaded)
- Plans to put HPC resources behind portal to deliver answers to questions too difficult to answer on a desktop with downloaded data
- Challenges:
    - petascale (2-10) management & replication
    - versioning, DOIs/citation
    - metrics, monitoring, and notification
    - more diverse data, extended metadata model
    - tension between R&D and production
- Q&A
    - Dan: diversity vs. community standard?
        - WMO came up with data formats for weather forecasting a long time ago
        - some things we can control & some not, e.g., new markup languages
    - Yin: In some fields, there are too many standards
        - Yes, it's a problem
        - ...but at least they tend to be rigorous in their definitions
    - Yin: Are there any mappings between standards?
        - Don has not been involved with this
    - Shantenu: How much of this work can be generalized to other communities (e.g. bio)? What lessons can be learned from your experiment?
        - must be assessed for each community
        - several groups are trying out. More next year.

Yanif Ahmad, Massive Parallel Agile Data Views

- http://www.dbtoaster.org/
- Applications:
    - algorithmic trading
    - algo simulation and backtesting
    - LHC (monitoring & analysis)
    - Google Caffeine

    - Google Crawler
- Scale-out on data size:
    - MapReduce
    - Communities build lightweight engines (trading of consistency for scalability)
- Scale-up on dataset evolution
    - stream, complex event engines
- Shantenu:
    - slide 1: one more axis for distributed data
    - usage of scale-up and scale-out
    - scale-out in 3DPAS refers to distribution of data (not data-intensiveness)
- Online Analytics in the Cloud:
    - DBToaster: dynamic data management system
    - Cumulus: an incremental, horizontally partitioned DDMS
- Applications:
    - high-frequency order book trading
    - output of scientific simulation
        - SIGMOD: Scientific Data Management at the Johns Hopkins Institute for Data Intensive Engineering and Science, http://damsel.cs.jhu.edu/papers/sigrec10-jhuidies.pdf
- Programming with Views: Materialized Views - a building block for analytics
    - Key concept: delta queries (for agile views)
- Cumulus: scale-out agile views via shared-nothing distribution and horizontal partitioning
- Programming models:
    - use views as shared, read-only data structures
    - perform local processing on top of views in parallel fashion
- Q&A:
    - Yin: related work bigtable (http://labs.google.com/papers/bigtable.html), pig latin (http://pig.apache.org/docs/r0.7.0/piglatin_ref1.html)?:
        - key/value store
        - programs still have to be written by user (not high-level query language)
    - Simon: similar to functional languages
        - it's actually a functional languages (e.g. map)
        - lazy evaluation is key

Dan Katz: Application Discussion

Application categories:

- get data from files
- get data from files & streams
- get files from streams

Applications (see also White Paper)

- Biosciences:
    - "data-intensive" workflows

- WLCG app:
    - no train model anymore
- VO app for astrophysics
    - no high requirements at this point
- Sensor network app
- Bioinformatics: metagenomics project
- Climate:
    - CDAT
    - Can also be migrated to OPeNDAP. Big data reduction when using OPeNDAP
    - Some indexing for efficient data access
    - some multi-site query support
- Astro(LSST)
- Oceanographic app
- Environmental data exploration
- Power grids
- Astro (cmb)
- Fusion (multi physics simulation codes)
    - Which group / project / research does this use case come from?

What is important in these applications?

- streams vs. files
- processing data as streams vs. processing data as archived data
    - can treat archived data as stream, cannot treat stream as archived data
- data trains vs. stream standing queries
    - move algorithm over data vs. move data over algorithm
    - What is a “data train”?
        - different groups at LHCG had different analysis code
        - code submitted by scientist is run by the “data train”
        - Make sure that data is read only once (for multiple analyses)
- file vs. databases
    - Are files typically structured? Does that enable DB to be replaced?
    - Dan’s mindset: Files read completely vs. DB queried for parts of it
    - J. Gray paper: Abstractions for scientific data: http://research.microsoft.com/en-us/um/people/gray/ (somewhere here) http://research.microsoft.com/pubs/64537/tr-2005-10.pdf
- What’s the difference between reasoning and querying?

3D axes: data size, ingest/refresh rate, processing rate

- others: diversity of data (how many types?), data lifetime
- data-intensive:
    - just doing something on big data versus
    - IO rate >>> CPU cycles (Amdahl number)

Adam B:

- extra category: data from services (workflows)
- mashing up dataset from different web services
- Shantenu: What is the fundamental difference here?
- Neil: this use case fits: distributed, dynamic (no control over service), data-intensive (according to Adam's definition IO rate >> cpu cycles)
    - TODO: Adam to describe app

Taxonomy/Application vectors:

- Jen: What taxonomy to use? There are already many taxonomies
- Shantenu: we have to find the vectors to construct our taxonomy
- Neil: We don't know yet, which information we want to collection from application owners

Target Audience for white paper:

- community
- NSF, DOE, funding agencies,...

Neil: social simulation (city simulation)

- how do city change over time?
    - TODO: Neil to describe app

Don: data diversification is a big issue

- impact if data is owned by different people (protectionism)
- language of a model is not understand by all scientists
- data integration problem
- also a big issue in biomedical domain (additional issue: data reduction)
    - only store diff to reference genome (99% never change)
    - diverse genomes, but a standard way to store a genome
    - no need to throw away data (if diffs used)

Data reduction:

- very community specific
- climate:
    - compression sufficient
    - don't throw anything away
- sky survey:
    - 1 TB data each day
    - 2 h download
- medical images:
    - TODO: Yike describe data reduction scenario for medical imaging

---

Day 2

Jennifer Schopf, Production Quality Data

Agenda:
How we build software?
How could we build data?
Underlying themes: reproducible sciences, organization requirements

neon: 150 TB/yr
LSST: 30 TB/yr
…
most interesting: long tail of small sciences

poor data practices:
information content decreases from: time of publication -> career changes -> retirement

How do we train data-intensive scientists?
http://www.nature.com/news/2010/101013/full/467775a.html

How do we build software?

- **Personal**
  - used by me
  - I am the end-user
  - I do all the coding
- **Prototype**
  - used by my group 5-10 people
  - I do most coding, but additional grad student
  - No testing besides use
- **Moving toward Production**
  - Coding by several folks
  - Testing by friends & end-users
  - some test cases
- **Production Software**
  - used by people I don't know
  - coding by larger group
    - common repository
    - agreed coding standards
    - real software architecture
  - formal testing
  - bug fixing process
  - formal release process
  - license

How does this relate to data?

- End users who aren't me
- multiple producers/collectors of data
- formal testing

- QA/QC, Bug fixes
- Documentation
- formal release process
- citation

What is data?

- observations
- data analysis result
- modeling results
- software / environment
- metadata
- data refers to everything needed to have reproducible sciences
    - Steve: does this exclude derived data?
        - not always
        - what is the difference?
        - to be discussed
    - we need the option to reproduce science. Often this is not possible

Who is using your data sets?

- This is all about sharing

Local Archive

- In SW-world this involves some kind of code-checkin to a repository
    - get a sanity check
- When data comes off an instrument, do a check

Metadata standards:

- Biologist rather use someone else toothbrush than someone's else metadata standard (C. Stewart, IU)
- Hundreds of ontologies to choose from
- Standards are complex, but a necessary starting point

Testing/QA

- make data reliable and useful
- additional QA:
    - check data ranges
    - correct instrument error
    - sometimes first derived data products
- need to accept bug reports
- Omer: Is correcting data actually a bad thing?
    - we need good metadata surrounding the dataset so that we understand those things

What data to store? Every year we produce more data than we can store?

Documentation

- story around not the data (not just metadata)
- how was the data collected? details on instruments, QA/QC

Discovery

- Make it findable
- remember the web before Google
- data on cards is not discoverable

Formal Release Process

- note “release” - not “publication”
- there are other models to make data available such as the sw reelase cycle. Alpha, pre-beta, beta, rc, release, revision, documentation, feedback

License

- Get credit for your work
- intellectual property issues
- creative commons license
- Jon: Reward structures: data archives are not as valuable as publications
- What is the chance that reward structures change?
    - software is not as rigorously cited as other papers

What about dynamic & distributed data?

- similar to dynamic or distributed systems and software:
    - you don’t own it
    - unreliable: access, changes, etc.
    - you don’t know when it will be updated
    - replication errors
- Lack of control: fact of life, how to compensate

Recap on building production data

- metadata - make it understandable
- local archives

Data Policies

- NSF data policies: Investigators are expected to share with other researchers, at no more than incremental cost an within a reasonable time, the primary data, samples, …
- As of Jan 2011 you have to have a data management plan
- other funding agencies have similar policies
- Jon: Data defined as reproducible science? No, NSF does not define data
- Other places have data requirements:
    - Science: All data necessary to understand article must be made available to readers of science
    - extended to compute code, supplementary materials to contain content to

methods and data descriptions
- How to review data?
    - at least: is it there?
    - example scheduling: some data is usually missing and thus, makes it difficult to review
    - to what level of detail do I have to record data in order to make it reproducible
- Data sharing vs. open access
    - open access not set at NSF (some opponents)
    - some universities committed to open access
    - Caltech researchers are not even allowed to open source software

Conclusion: We should think of data as infrastructure
Data is a first step toward reproducible sciences
treat data like infrastructure
plan for it to be here longer than you expect

Toronto agreement: data shared and can be mined after embargo period:
- http://www.nature.com/nature/journal/v461/n7261/full/461168a.html

Yikes: What is the standard for capturing experiment data?
- Argue on the fine line: What do we need to capture in order to reproduce the experiment
- depends on the field

Neil: Discussion
- What are things that make data accessible, understandable and useful?
- How are these affected by the characteristics we are trying
- What's the place for derived data?
- How to review data?
- How to do QA for very dynamic data?
- Can we do the same as Alzheimer's community in terms of achieving more impact through more sharing?
- What are standards for capturing data?
- When the applications can be mapped, can we see what the missing pieces are?
- What are the important characteristics for enabling reproducible sciences? (Dan: can we answer this in 3DPAS? Neil: Are there certain programming abstractions that make it easier to do reproducible sciences?)

Yike Guo: WikiAnalytics
- Genome sequencing is cheap
- find cancer gene, impact on family tree
- how to find the right people that can analyze a genome sequence?
- requires a collaborative authoring environment as Wikipedia
- no one alone can annotate the entire genome
- Book: Wikinomics

- example: cloud computing wikipedia item:
    - created Sep 4, 2007
    - period of chaos
- Collective intelligence and social computing
- how to
- cluster and annotate pictures?
    - gwap: http://www.gwap.com/gwap/
    - game for collective annotation of images
    - people like it because its fun (incentive model is important for success)
- Wikianalytics: analysts work collectively in knowledge discovery
- Question: Quality of annotations?
- Single Data Multiple Models for improving accuracy
- Multiple Data Single Model
- Multiple Data Multiple Model - heterogeneous
- WikiAnalytics is far beyond these models:
    - incremental learning
    - human is key provider of prior knowledge
    - KM System (MediaWiki), Social Network (Facebook), Cloud (IC Cloud)
- Imperial Cloud:
    - departments can plug in their machines that become part of the cloud
    - Custom code on top of Xen
- Back-end of WikiAnalytics
    - create workflow applications with Inforsense
    - share apps on Facebook
    - U-BioPred - MediaWiki for genomic information
    - similarities to Taverna / MyExperiment

Infrastructure Discussion

- How do we bring infrastructure and applications together?
- Automated vs. crowd sourcing techniques?
- Why does Wikipedia need an auditing process?
- For non-controversial articles Wikipedia isn't much worse than encyclopaedias
- see Neil's notes

Adam Barker, 3D Programming Abstractions and Systems

Two projects:

1. Abstracting cloud resources from research
2. Model of coordination to reduce data transfer in data-intensive applications

Elastic Virtual Infrastructure for Research Applications (ELVIRA)

IaaS for Research

- simple for researchers, e.g. EC2

- Demo of complex steps necessary to boot up an EC2 instance
- Elastic Wrapper - abstracted gateway to cloud resources
    - collaboration between multiple sites
    - management of resource usage
    - scientific workflows
- How to configure your instance?
    - set of pre-defined instances
    - install necessary software on any instance on-the-fly
- How to control costs?

Elastic Wrapper:
- standards: JSDL as interface

Example applications:
- GAP - computational algebra:
    - multiple GAP servers that communicate via SCSCP protocol
- Astrophysics application

Research question:
- Cost modelling
- interface specification?
    - provide a generic interface for domain scientists? SAGA?
- Autonomic resource management

Optimising Data Transfer in SOA
- to efficiently execute large-scale, data-intensive applications one must take into account the volume and pattern of communication
- Standard WSDL services, e.g. that can be access e.g. by Taverna workflows
- Workflows components become a valuable asset that can be traded
- Models of service coordination
- reduce data flow in distributed components, particularly SOA
- Orchestration:
    - Workflow Orchestration engine
    - pulls data and pass it to Hadoop cloud
- Assumptions
    - re-use existing pieces of technology
    - services are owned and maintained by different organisations
- Hybrid orchestration:
    - using proxy to invoke services, e.g. pull data or execute an Hadoop job
- General architecture - not limited to Web Services
- Reducing data flow accelerates application
- Experiments on planet lab
- Proxy network:
    - Caching

    - Shim output to a new input
- Web Service Choreography:
    - decentralised service coordination
    - optimal since no central control
    - WS CDL, http://www.w3.org/TR/ws-cdl-10/
    - Specification
    - Model checking
- Conclusion:
    - Abstraction are vital for domain scientists
    - Data-intensive applications can be bottlenecked with data movement

Discussion:

- Which abstractions are vital?
    - Taverna
    - Tools vs. Abstractions
    - Coordination abstractions/models used?
        - central, decentral, hybrid
- What scientific application will benefit from optimal data movement framework?
    - DAG-based applications
    - Is there a DAG application that uses multiple distributed resources?
    - Taverna:
        - 10-20 widely distributed services
        - mashup of different data sources
    - criteria: data-transfer per computing

Roy Sterrit: Autonomic and Apoptotic Systems for Distributed Dynamic Data

- inspired by human body's autonomic nervous system
- self-management, self-configuration, self-optimisation, self-healing, self-protection, self-*…
- managed component, autonomic manager (self-monitor, self-adjuster, knowledge, adapter, environmental monitor, AM <-> AM Comms (Reflex Signal)
- Autonomic manager can be considered as stationary agent
- Autonomic communication channel
- sensors <-> effectors
- Heartbeat Monitoring (HBM) - I am Alive extended to Pulse Monitoring (PBM) - I am (un/)healthy
- Autonomic Communication:
    - Knowledge Plane
    - MGT/CTL Plane
    - Data Plane
    - Novel Communication Paradigms for 2020 on Autonomic Communication
- Example: Forecast and automatic provisioning of servers based on BOPS prediction
- User Case:
    - Virtual Archive
    - High Energy Physics

- 3Ds:
    - AC provide self-managing environment for 3Ds
    - Managed component (MC) could be managed data and if so what are the properties of the Autonomic Manager (AM)
    - Application? Autonomic/Self-Mging Env creating substantial self-managing 3D. (yet non scientific application). Expertise from community
- Apoptotic Computing
    - another biological metaphor
    - preprogrammed death by default
    - Apoptotic 3Ds? (=data pollution?)
        - ethics, security trust - protecting sensetive data
    - http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=05688151
- Q&A
    - Use of autonomic techniques for placement of data survey paper? not known

Simon Dobson, Discussion Points

- Data models:
    - metadata, error models, instruments metadata, provenance
- Programming abstractions
- Querying vs. programming vs. reasoning
    - Queries: get the data you want
    - Prog: process entire dataset completely
    - Reasoning: use knowledge to complete/generalise/specialise
- Making reasoning explicit allows higher order analysis of what's happening
- Separate patterns of data access: separate reads/writes
    - leverage the rates at which different operations happen
    - use diffs to track changes of a basic structure (capture dynamics with that data evolves)
- Metadata:
    - Should data be corrected?
    - Is error model integrated into the sensing process in an irremovable way?
    - Should we publish the raw dataset and the model?
    - version dataset into raw and raw
- Data as software: data engineering process that mirror SWE process
- lifecycle: policy, drm for data
- data as a managed component
    - control & adaptation as an attribute of a dataset
- Are there programming abstractions that make things better:
    - keep metadata through the processing chain
    - workflow?
    - just wrappers ontop of a stable core?
        - e.g. web service ontop of fortran code
        - does not change data interaction

    - keep data neutral and don't wrap into "object" abstraction:
        - oil / water discovery on top of same data
- Aspects:
    - Complexity of data
        - metadata, processes, lifecycle
        - typing and bundling of multiple elements
    - Patterns of processing
        - seq, random, structured, trains, …
        - query, whole (file), reasoned, …
    - Composition
        - Orchestrate, choreograph, workflow

Discussion:

- Shantenu: programming the infrastructure not the application
- application often stays in an immutable state
- notion of a workflow
- workflow have provenance built in
- Streaming:
    - very many workflow tools, but very few support streaming
    - streaming is a lot of streaming by convenience instead of necessity
    - functional programming is a kind of streaming
    - LHC streams data (but filters it to handle these large amounts of data)
- Why don't we see more autonomic/workflow capabilities on top of existing infrastructures, e.g. ESGF?
    - Don: generally not on prod. infrastructures
    - Shantenu: workflows could utilise autonomic services
    - automatic decision making == autonomic?
        - adaptivity is the key distinguishing feature
        - higher-level goals that drive decisions
    - P2P approaches possible, e.g. for file sharing
    - Communication "people" have a better experience in autonomic optimizations
    - What can fail in real-life: Email, IDS, domino's bring down the data center (supercomputer brings mail server down)
- Treat everything as managed component?
- What would be possible forward corrective measures instead of backward recovery actions?
- Fault Detection and continuous testing:
    - Condor runs test-jobs
    - Basic service monitoring: pings
    - NMI
- Is FT a programming system issue?
    - Omer: can become an application-level concern, e.g. provide execution alternatives for workflow
- Paxos if you want to make system really reliable => very expensive

- How many replicas do you need to deal with n failures? Where should this replicas placed? Affinity between replicas and compute?
- Why do science application don't take this account?
    - Media streaming error model is well understood
    - Apps use checkpointing
- We can't even manage compute-intensive jobs, what chance do we have to do data-intensive job?
    - treat sensor data similar to streaming data?
    - sensor characteristics must be better understood
    - What are the classes of application / what are the application characteristics where such models can be used?

Omer: Infrastructure

- What are the 3D aspects in ingest, analytics, recording (preservation) applications?
- Can autonomic approaches enable adaptation of existing workflows
- see Omer's slides
- Shantenu: Which attributes arise from distribute? which from dynamic?

Day 3

Infrastructure Discussion

DPA Vectors:

- Execution Unit
- Communication
- Coordination
- Execution Environment

What about 3D? Does every app have value in each vector?

- if value for an app is 0, is app not important for 3DPAS?
- Amdahl number as a characteristic?
- Other characteristics:
    - Data ingest
    - Analytics - What is the product?
        - knowledge
        - information
        - derivatives you wish to share
        - infrastructure:
            - how many VMs?
            - move data to compute or vice versa?
    - Post-Processing/Reuse

Sensor Network

- Data ingest not part of application (it's part of the system)? Infrastructure vs. application

Dynamic "Data"
Which we define as having particular factors varying in time and either the application or system or both adapting its behaviour after the point of deployment based on this variation

- The volume of data is changing
- There is variability in the rate of data ingest / jitter
- The structure of the data is changing
- The input / output sources are changing
- The queries on the data are changing
- The execution units are changing
- The "application pipeline / workflow" is changing (e.g. steering, adaptive computation)

Some example application categories that are inherently dynamic include: adaptive sensing, evolutionary computation.

Distributed

- control,
- resilient / redundancy
- not all in same place

DPA:

- Coordination
- Communication
- Execution Environment

Data-Intensive:

- Amdahl numbers:
    - Compute / delta data transfer (the amount of data that could be worked on)
    - Compute / delta data transformed (the amount of data worked on)
- Rate of data ingest
- large datasets
- large, complex queries
- concurrent access
- compute / data transferred
- compute / data transformation

What are we giving to the outside world?

- application
- application vectors
- missing parts in the infrastructure

Scope of this document

There are multiple “D”s that shape the landscape of the applications that are of interest. However, we will take data-driven/oriented/intensive as a fundamental attribute when defining the scope of applications and analyzing the associated infrastructure.

Don and Dan’s part – Climate/ESGF

Aimed at supporting international CMIP/IPCC intercomparison activity.

Modeling:

- Climate centers run prescribed set of common experiments; produce 2-10 PB of data
- Centers can publish their own output data or send it to another center to publish
- data generated over ~2 year period (then post-processed and published over a few more months)
- Properties: **data-intensive**, **distributed** centers, lots of computing

Infrastructure/system:

- ESGF develops and deploys a federated network of gateways and associated data nodes
- As models run at each center, output data is post-processed into packages with common formats and prescribed metadata conventions
- Most centers will deploy data node software stack, and using this, they manage the data from their experiments
    - the data node software stack scans the data, makes sure it has the right metadata fields, does QA/QC, build a set of catalogs of the prepared data
        - minimally, catalog provides HTTP links to data elements
        - also can provide GridFTP endpoints
        - also can provide product services that abstract the dataset in other ways - get whole file, subset, browse, etc.
- When the center is happy with the data/catalog in the data node, they publish it to a host/affiliated gateway
    - this submits the catalog to the gateway
    - them the gateway shares this catalog with other gateways so that all gateways have a consistent view of all the published data
- Core archive (1-2 PB, most popular data)
    - will be replicated at several sites
    - replication activity manually requested/initiated by a gateway owner
    - replication: copies catalog to replica site, do data transfer to new data node, then publish catalog, push catalog to (this) gateway associated with new data node
- Currently building notification service
    - track which users have downloaded data - if a problem is found in a data set or if new data is added to a data set, notify users who have downloaded/replicated it
- Properties: **data-intensive**, **distributed** gateways and data nodes, data appears in the system over time (**dynamic**)

Application:

- User can:
    - browse/search a catalog at any gateway and locate data, which might be hosted by a data node affiliated with another gateway. (can output a wget script that can later fetch the data)
    - authenticate, gain access to a group
    - download data via http or GridFTP or access product services (uses data retrieval syntax (DRS) - can be scripted)
- abstractions that can be used by a programmer: data is stored in NetCDF format, uses CF metadata conventions, can be accessed using DRS
- Some users will analyze data from a single model
- Many applications are multi-model analyses - many users want to look at the same parts of the output of some/all of the models, to understand if and how the models differ
- Some centers will gather some/all of the core archive (plus more, perhaps) on local systems for local users to perform “power analysis”

Understanding Dynamic Data:

There is a general appreciation that the challenges of data-intensive applications span beyond large-volumes and management/curation. One specific attribute that we believe is pervasive but not necessarily obvious is “dynamic data”. The belief is based upon the fact that data is often distributed, and as a consequence many attributes of distributed systems permeate to data. Additionally the requirement of many applications impose a requirement that data is dynamic.
Some example of such dynamism:
- Change of data operated upon.
- Variability of data rate -- arrival, consumption or ability to generate (due to dependencies)
- Data placement and scheduling issues are time dependent
- The nature of coordination between the different execution units changes

In the section on applications, we will discuss how and which of these different scenarios are encountered.

Data-Intensive Applications:

- Amdahl’s number has been useful in characterising “static” data-intensive applications and systems. We believe there is scope to generalize Amdahl’s number to capture attributes of dynamic data applications.

$\alpha 1$ = Num. of Compute Operations/Amount of Data Operated or Consumed
$\alpha 2$ = Num. of Compute Operations/Amount of Data Transferred (across network)
$\alpha 3$= Num. of Compute Operations/Amount of Data Transformed
$\alpha 4$= Num. of Compute Operations/Amount of Data Change of ??

$\beta 1$= Amount of Data / Degree of Distribution
$\beta 2$= ??

Jon's Part - Mechanisms and Metrics
In this section, identify existing mechanisms that operate within
DDD (or subset thereof) infrastructures and metrics that describe their behavior

- dynamic data/computation:
    - proxies/in-network data capture, routing, in-situ or in-transit processing
    - metrics: low latency data access, bandwidth reduction
- distributed data/computation:
    - nebula cloud:
    - metrics: elasticity, harvest = amount of data/complete data
- data-intensive:
    - hadoop
    - metrics:
- life-cycle (data-ingest, analytics, post-analytics):
    - TBA

Neil's part - Vectors
Vectors (Axes) for consideration for applications (Shantenu, Jon, Omer, Neil, Simon, and Dan)

- Used to differentiate between different kinds of dynamic data
- Ratio of compute:data volumes (use this as a scale to place these applications)
- Timeline to indicate how applications fit in on a time axis in terms of the dynamicity of the generated data
- Data set size (unit of granularity) to support harvesting of data – files, portions of files, data elements, grouping of these, etc
- Data availability – data not ready when you start the analysis (incomplete processing) – either: (i) it could be, but is not; (ii) could never be.

Provide introduction to why we are considering the vectors / characteristics
Go through the applications and attempt to "tag" them with characteristics leading to a generated folksonomy of vectors.

https://spreadsheets1.google.com/ccc?hl=en&key=tWTnLK8ZFglMOzua8hwFFMA&hl=en#gid=0

Map the applications and characteristics (in a table?) to identify clusters.
Understand if the clusters correspond to use of particular architectural / model abstractions (e.g., stream processing, MapReduce).
Understand if the clusters correspond to a particular requirement on the underlying infrastructure (link to Omer's work)

Omer’s part - Cyberinfrastructure section in white paper

- rename to Infrastructure
- Differentiation between application requirements
- Show instances of applications running on infrastructure, what does it do, what does it provide - for each of the stages: data ingest, analytics, post-processing/managing reuse / derivatives sharing
- Have a table of applications, identify what the infrastructure provides at each stage based on what people are actually using; then repeat for ideal scenario, what would they like to have? Summary table covering all applications
- Identify areas of “missing” infrastructure where there’s a cluster of ideals not being fulfilled by what’s currently available