3DPAS-Meeting-1: 01 and 02 November, 2010

Dynamic Distributed Data-intensive Programming Abstractions and Systems

#3DPAS

Introduction of participants:

1. Silvia Olabarriaga, Academic Medical Hospital of Univ of Amsterdam
2. Martin Kersten
3. Stephen Pascoe (in lieu of Bryan Lawrence)
4. Adam Carter, EPCC
5. Steve Fisher
6. Simon Dobson
7. Omer Rana, Cardiff
8. Dan Katz
9. Bob Mann
10. Neil Chue Hong
11. Andre Luckow
12. Jon Weissman
13. Jon Blower [Remote]
14. Shantenu Jha

Shantenu Jha:
Dynamic, Distributed, Data-Intensive Programming Abstraction & System (3DPAS)

- Follow-up theme to DPA
- focus not only on data-intensive, but in particular on dynamic and distributed data
    - Infrastructure can be dynamic!
- Three tracks:
    - Applications
    - Programming Abstractions and Systems
    - Cyberinfrastructure
- Aims:
    - Survey domain of 3DPAS
    - 3x3 matrix (3 tracks x DDD)
    - Identify interesting problems (short term) versus research challenges (long-term)
    - Can we find areas to take theory into practise?
- Related Theme DIR: http://www.esi.ac.uk/files/esi/Theme15-proposal.pdf

Jon Blower, Reading
Application Track

- What kinds of application?
    - applications limited in some way by concerns of data (big, diverse, dynamic, distributed)
- Big Data:

    - storage costs
    - biggest issue: moving data
    - mitigation strategies: compression, subsetting
- Diversities:
    - different file formats, metadta, acess mechanisms (ftp, web service), access policies (e.g. European data policies are very strict)
    - fault-tolerance
    - mitigation strategies:
        - standardization is a very slow process
        - virtualization
        - sso
- Dynamic data:
    - e.g. sensor data
    - unpredictable data rates
    - quality control
    - mitigation: automated (pre-) processing of data in-flight
- How to attack these problems?
    - Infrastructure: What kind of problems are generic between the different domain areas? Support the interactive access to data
    - Algorithms: new algorithms that are optimized for new infrastructure
    - Programming abstractions
    - good quality software
- Questions:
    - How would you classify your data problems?
        - Big/dynamic/distributed/diverse
        - batch or interactive
    - What kind of solutions are you looking for?
    - What would your ideal infrastructure look like?
    - What problems could you attack with {more contacts, £100k, £500k}?

(e-)Infrastructure for (e-)BioScience
Silvia Delgado Olabarriaga

- e-BioScience Group (@AMC) introduction
- close collaboration with Medical imaging, bio informatics, dutch grid and informatics
- AMC Research IT Today:
    - used to be able to conduct research on a single machine
    - now: you need to buy a cluster to conduct research
    - clusters are not shared
- e-BioInfrastructure gateway as central access point to resources:
    - enable and enhance biomedical research at AMC:
        - just received first request for a non-research, clinical application
    - facilitate access to (Dutch Grid) resources
    - services and support
- Generic services:

    - gLite middleware
    - Web services (SOAP)
    - Moteur workflow management system
    - DIANE pilot job framework
    - Virtual resource browser (GUI for grid-enabled protocols)
    - Example of executing a workflow in the grid
    - Workflow can now be triggered from a Web application
- Ecosystem:
    - What exactly is a VO? Sequencing facilities can be very distributed
    - Link between Sequencing facility and research laboratories is very demanding:
        - Omar: What data rates do you see from the sequencing facility? (later)
- Applications:
    - Medical imaging: functional MRI
    - Sequencing
- Next generation sequencing:
    - 3 different kinds of sequencing machines
    - Current data for one run on Roche 454 (2005-now):
        - 7,5 hours
        - 1,000,000 sequences
        - 500 nt per sequence
        - 35 GB data (including images)
        - 500 MB data (excluding images)
        - = 500,000,000 bases
    - Current data for one run on ABI SoLiD (2005-now):
        - 3-5 days
        - 150,000,00 sequences
        - 50-100 nt per sequence
        - 2-4 TB data (raw data) = 15,000,000,000 bases
    - Basic sequencing process: Wash - Sequence - Align
        - parts of the process well parallelizable
        - used by many different applications
- Analysis software:
    - very “hacking” style - makes support very difficult
    - BLAT, BLAST, BWA
    - Roche
    - R
    - Programming language: Perl, Shell, Java
    - Moteur
    - ~314 different software components
- Example Workflows/Scenarios:
    - Alternative Splicing:
        - Grid for all-pairs computing
    - Alignment
    - BWA:

        - Typical data amounts for intermediate files:
            - input 25-35 GB
            - quality files: 50-80 GB
            - Reference DB: - 3.2 GB human genome, 140 MB one chromosome
            - Reference BMW index 4.5 GB
    - Challenge: data locality often not taken into account
- Barbera's nightmares:
    - Learn to program in Java
    - copy large data centers
    - how to use cloud computing
    - other people have similar problems (Map of al sequencers in the world)
    - people still work with very primitive tools despite having this expensive machine
- Data distribution:
    - granularity
    - programming
    - location (affinity: bring data to compute or compute to data)
    - management
    - 30% of the errors come through failed data transfers
    - gLite middleware has some issues in this area
    - problems cannot be solved by solely throwing hardware (clusters) on it
    - low awareness for "grid sharing thing"
- Data security:
    - DNA data has high risk of re-identification
    - Whole genome data is not anonymous (ever)
- Data transport:
    - Acquisition => data server
    - How to bring the data somewhere else? All kinds of "hacked" solutions, e.g. staging servers
    - Time to server setup is very long: server has been bought 3 months ago, but is still not completely setup (in the DMZ)
- Error handling:
    - if any transfer fails it takes a long time to find out what happened
    - different variants of data transfer: pre-staging, staging after job has started
- How to build the system?
- Summary:
    - data analysis process is different every time
    - scaling data analysis is very costly
        - people who do this lack expertise and tools
    - scaling data transfer is unknown at this moment:
        - Omar: What are the issues? Network bandwidth, Jitter,...
            - Using many (small) files is a potential problem
            - 100% of data is stored in files, 0% in databases
            - planning on how to use the resources

            - no compression at the moment
    - main focus is on results not in the process
    - e-Infrastructure:
        - needs more powerful generic services:
            - data, computing, monitoring, security, error handling
        - Too many (incomplete) alternatives to choose from
        - Interoperability is lacking (to integrate pieces)
    - Classification:
        - Big/Dynamic/Distributed:
            - Data is big for the users
            - not dynamic
            - not distributed
            - diverse
            - data is too large to store it at the local site in the future
        - Batch vs. interactive:
            - calculations are done in batch
            - result analysis should be interactive (currently offline)
        - What are we looking for?
            - “help Barbera”
            - improve functionality of generic services
            - improve implementation of generic services
            - tool that helps researchers to create generic services
            - the “ideal” diagram shows a single (“blue”) layer (a “platform”?) between all user-facing parts and underlying (“red”) infrastructure
            - easy-to-use, operate and maintain
            - self-*: self-configuration, self-adaptive
            - reliable
            - fault tolerant
            - researchers “don’t look for the source of errors; they just rerun the workflow until they get all the results”
        - No funding but more contacts
        - £100k: data transport problem
        - £500k:
            - data distribution mechanisms combined/integrated workflows
            - dynamic self-adapt to execution conditions for resource usage optimization, security and fault tolerance.
        - Some important “dynamic data” is the feedback from the systems/ infrastructure as to e.g. the success of data transfers
    - Discussion:
        - Do the errors only occur in a WAN setting, but also local? It’s currently too difficult to determine which data should be pre-staged to local systems. Requires that data are split etc.
        - Available resource information is insufficient
            - e.g. gLite can’t always show available space on a storage element

(it's meant to, but it's not reliable)
    - CS/IT know-how in research group usually limited

Cyberinfrastructure Challenges
Stephen Pascoe, Bryan Lawrence

- Center of environmental data archival (CEDA)
- Simulation Data Deluge: data is getting more voluminous
- CMIP5 is a global problem: simulation data is generated globally
- Not only model output, but also various sensors and observatories
- in 2020 we are talking about 20,000 TB of data
    - Google/Ebay handles 20 PB in a cluster of ~100 machines
- Storage can't keep up:
    - not only the pure volume, but also the problem of getting to the data
- 3DPAS:
    - Data-intensive
    - Not dynamic
    - Distributed

    - Applications
    - Programming Abstractions & Systems
    - Cyberinfrastructure
- CMIP5 Numbers:
    - 17 institutions
    - various different models
    - institutions requested 2.2 PB
    - we can handle ~1 PB (replicated)
    - Definition Replicated: We receive a set of data files, which we copy to multiple locations using GridFTP.
- ESGF Architecture Innovations:
    - versioned data
    - replicated data
    - federated access & security (OpenID)
    - quality control
    - multiple metadata streams
    - Architecture:
        - Gateway Nodes (Web applications):
            - Gateways are aware which data they are managing
            - Will send users direct links or a download script (wget)
        - Data Nodes
        - Federation & Replication:
            - Various institutions run a gateway and data nodes
                - e.g. PCMDI
            - Every data node has a section for replicated, requested and optional data

            - 10 GB/s network
            - Shipping 16 TB disks (backup plan)
        - Identity & Versioning:
            - Data reference syntax: controlled vocabularies encoded as ESG dataset ids, urls, paths
                - difficult to get the buy in from the participating institutions
            - Tracking Ids
            - Md5 for integrity checks
            - DOIs for some parts of the requested data
            - DRS include a version => multiple versions should be available simultaneously
        - Data Ingestion:
            - User produces a model which is processed by CMOR
            - Institutions must fill out a questionnaire which is published by the gateway
            - Common meta-data model is required
        - Quality control
            - data is checked for various levels of correctness
            - suite of automated checks (not sure whether sufficient computing resources to run these are available)
- Technology Issues:
    - Users want to interact with portals and applications (potential/casual users prefer easy-to-use interfaces, e.g. an Excel spreadsheet). Programmatic access is often required by sophisticated users.
    - Access rights driven by metadata
    - Getting metadata is hard
- National problem:
    - in the future data must be closer to the user. Local analysis cloud.
    - getting good bandwidth out of 10G networks is difficult
    - public clouds are too expensive due to data transfer costs
- Social challenges:
    - rewards (citations, curation)
    - global cooperation and planning (more international reliance on each others infrastructure required)
    - global trans-institutional trust & reliance
    - licenses & ipr
- Infrastructure Genie
    - derived products on demand
    - cloud analysis platform that allows users to run their application close to the data
    - reliable global >1Gbs network + expertise to operate them
    - standardized data identity & version management:
        - need a version management tool that is easy to deploy and can scale to 100GB files
- Pragmatic goals:

    - Technology:
        - Contacts: idm, best practice, better networks, expertise
        - ~1PTE Big Data DVM
        - 100K: derive product applications
    - Infrastructure:
        - 100K: Cloud analysis testbed
    - Sociology:
        - ~1PTE: international project management
        - 100K: world domination
- Discussion:
    - How is data from sensors handled?
        - De-coupled system: Data is pre-recorded somewhere else and then shipped to the centre. Partners can designed how/whether to use the ingestion system to forward data to centre.
    - Limitation of network access to cloud: Does this issue also apply to other academic institutes?
        - in particular in international settings network limitations (e.g. certain routers) exist
    - Awareness for data pollution must be raised
        - “All data that isn’t used is just garbage”
    - Who decides which data gets removed?
        - depending on what we think is scientific important
        - some think: once data is easy enough (and cheap enough) to recreate, it no longer needs to be stored

3DPAS and (mainly optical/near infrared) Survey Astronomy
Bob Mann, Wide-Field Astronomy Unit, School of Physics & Astronomy, University of Edinburgh

- observational astronomy: over all of the electromagnetic spectrum (this talk focuses mainly on the visible light area, some IR and some UV)
- to understand everything you need the data of all parts of the spectrum
- old vs. new style:
    - single researcher vs. large team of researchers
    - many small programmes vs. few large surveys
    - target specific objects vs. map large areas of sky
    - manual data reduction vs. automated pipelines
    - data ends up in astronomer’s desk drawer vs. data ends up in a queryable database
- What is driving these changes:
    - you don’t have to be an expert of the instruments you use
    - software & archive are important instruments
    - Economics: “more science per night of telescope time”
    - Technology: detectors capable of higher throughput. IT can handle the resultant higher data rates
- Astronomical discovery space: Area - Depth - Wavelength - temporal resolution - angular

resolution - polarization
    - different science goals require coverage of different regions of this space
    - surveys covering a large region of this space can address more
- 3 generations of sky surveys:
    - photographic era (1950-2000): ~60 yrs observation time, ~10 yrs of digitisation, ~20TB of image data
    - first born-digital era (1995-2015): ~20 TB of image data per year
    - synoptic era (2009-2013): ~20TB of image data per night for a decade
- UK Infrared Deep Sky Survey (UKIDSS): ~1/6 of sky using wide field camera on UK infrared telescope in Hawaii (Mauna Kea)
    - “Deep” implies long exposure - can capture things of lower brightness
    - One pipeline runs on the mountain (instrument health)
    - data is written to LTO tape and couriered to Cambridge weekly
    - Cambridge: data reduction pipeline: ~100GB/night (remove instrumental signatures, combine images, detect and classify objects, calibrate positions & fluxes)
    - Edinburgh:
        - ingest data from Cambridge: catalogues into RDBMS, image metadata into RDBMS, images on disk
        - combine data from multiple nights
        - once a year: prepare release databases for WFCAM science archive
    - Users worldwide
- Data processing:
    - there has traditionally been and currently is a split between data reduction (in data centre) and data analysis (on user’s desktop)
    - all kinds of software products/programming languages are used for analysis, both astronomy-specific (IRAF, Aladin, TOPCAT) and general (Fortran, C/C++, Python, Java, IDL)
- Much astronomy is now multi-wavelength, i.e. they need data from distributed archives
    - develop infrastructure to support this
    - international virtual observatory alliance (IVOA, ~20 partners, W3C-like standardization process)
- IVOA introduces standard protocols for interacting with different kinds of archives. All protocols are registered at a standard registry (AstroGrid Service)
- Standard means for wrapping applications into web services
- Clients can interact with backend via VODesktop and AstroGrid runtime (also scriptable)
- All AstroGrid code is written in Java and is open source.
- Web services increasingly RESTful
- Use IVOA standards where they exist
- Many services run asynchronously (can easily be embedded in worfklows)
- VO in 2010:
    - basic system in place
    - reasonable set of standards
    - little attention to robustness & scalability

    - gaps: distributed queries across multiple datasets (currently looking into OGSA DAI) - demonstrated through the use of DQP over TAP (using UKIDSS and CAOM).
    - TAP over DQP over TAP: Method for implementing distributed queries without data centres knowing.
- Future directions:
    - more computation inside the data center (e.g. can't download ~5PB database at the end of a 10 yr survey)
    - data processing requirement getting worse:
        - LSST can't perform eyeball quality control
        - SKA (radio facility) - pipeline will need the most powerful computer in 2020
    - Analysis getting more time-critical:
        - LSST want < 1 min turnaround for detection of transient events - from a data stream running at 4 Gb/s
- Summary:
    - Data:
        - big
        - distributed
        - sometimes dynamic and diverse
    - Batch vs. interactive:
        - batch mode
        - some interactive analysis
        - astronomers don't expect instantaneous response
    - What kinds of solutions?
        - existing VO standards provide a good basis for supporting multi-wavelength analysis, but need robust, scalable implementations
        - future data processing will need high spec system
        - not clear how to put more of the analysis to dc
    - What would the ideal infrastructure look like?
        - don't know and don't care
        - want to run data analysis scripts in IDL or Python
        - must be open source
    - What problems would you attack?
        - no money:
            - understand where MapReduce could fit into all this
            - understand where we can replace existing AstroGrid code with a more generic middleware
        - 100k:
            - get large-scale distributed queries to run effectively
            - replace one (or two) AstroGrid components with a more generic middleware
        - 500k:
            - seriously address how a data centre can support the necessary

range of data analysis operations that would be required for, say, Euclid or LSST
    - extend OGSA-DAI/Admire framework to provide a robust, scalable system for mining distributed data
- Discussion
    - OGSA DAI looks promising. Some issues with join-queries.
    - Are (private) clouds part of the solution?
        - haven't heard a lot about experiments in clouds
        - JHU has done some experiments with Amazon, but had issues with moving the data to there
    - Sylvia's urgent computing for urgent care example
        - analysis of aneurysms
        - need results within 10 minutes of when the patient arrives
    - Shantenu's summary:
        - There are infrastructure issues (file transfer, middleware)
        - Infrastructure should be programmable and responsive, and most importantly, should work reliably
        - None of the commercial infrastructures were sufficiently usable either
    - Standardization:
        - Bob: hard to get an agreement on standards
        - Astronomy community has standardized on various file formats (e.g. for observations across different wavelengths)
        - almost no standardization (of file formats, analysis tools, etc.) in biological/medical community

Some General Discussion

Dealing with large number of data streams vs. large scale bulk transfers of data (comparing the "Google" model vs. model from Earth Observation data).

Identify types (ranges) of analysis that needs to be undertaken on data -- and identify subsequent impact on infrastructure that supports such analysis.

Afternoon, day 1

Infrastructure: Big Questions and Scope
Omer Rana, Dan Katz

- Two motivation themes:

    - identifying the type of coupling between data sensors (acquisition & creation) and subsequent data analysis (or simulation)
    - mechanism for supporting in-situ analysis at point of data capture (includes decision making to decide where analysis is done)
- Definition sensor:
    - sensor can include devices for measurement and monitoring of physical phenomenon (temperature, pressure, HBA1c, oxygen concentration in blood):
        - RFID tags, android phones, pulse oximetry sensor
    - some sensor types maybe application-specific
    - synonym: instruments
    - “something that generates data”
- Thema 1: Coupling
    - feedback loop between data capture & subsequent analysis (relates to NSF DDDAS)
    - better coupling between sensing and analysis
    - what to record:
        - which sensor to involve
        - frequency, duration of recording
        - accounting for connectivity to/from sensor
    - Coupling
        - energy profiles of sensors
        - data quality
        - self correction
        - location of sensor
        - data generation may be in bulk vs. streamed
        - data acquisition: people (w/ mobile phone), biomarkers, physical electronic sensors, event triggers
        - storing algorithms that generated data (repeatability) => provide e.g. pre-built VMs
    - Sensor Examples:
        - Embedded network sensing application:
            - micro-sensors, on-board processing, wireless interfaces feasible at very small scale
        - Biosensors (from a drop of blood)
        - multi-electrode oxygen concentration biosensor (able to detect NO and O2 emission from cells)
        - mappiness.org.uk

- Theme 2: in-situ analysis
    - QoS issues associated with data transfer and use

Jon Weissman:
Clouds

- New cloud infrastructures

- Motivation:
    - data is increasingly large
    - think: multi-d scientific datasets, images, video
    - data is increasingly distributed (internet-scale, users, experiments)
    - data is intermittent, bursty
- One solution:
    - the central cloud:
        - pour all data in
        - sprinkle in some algorithms
        - results out
        - no limits on storage & computing
        - appealing to 3DPAS: on-demand is very attractive
        - but, there are short-comings
    - But, dynamic data requires a distributed solution:
        - low latency: when data shows up, compute on it, near the data
        - current centralized cloud model is high latency: both deployment and access
        - not limited to one cloud - there are different clouds
- Potential bottlenecks:
    - data source <-> cloud
    - user <-> cloud
    - cloud <-> cloud
- Idea: Make cloud more distributed
    - move it closer to data, end-users, other clouds
    - and dynamic: reduced latency (responsive), cost efficient
    - How? exploit the rich collection of edge computers
    - issues associated with edge computers:
        - e.g. bandwidth throttling
        - constraint on what type of edge computers to utilize
- 2 projects:
    - augment cloud with data-centric proxy
        - introduce a set of proxy nodes
        - goal: accelerate data transfer between user and clouds, cloud-to-cloud, source and control
        - overlay network on top of an existing network
        - Roles of proxies:
            - cloud service interaction
            - routing
            - caching
            - computation
    - re-factor cloud to a more distributed, dynamic platform
- Application example: Montage
    - accelerated at three different points
    - acceleration in data transfer from SkySurvey to compute nodes

    - acceleration of various inter-stage communication
- Dynamic ubiquitous cloud: Nebula
    - decentralized, less-managed cloud
        - application hosting platform
        - dispersed storage/compute resources
        - low user costs
        - Dan: A main argument for cloud is the economy of scale achieved by centralizing resources.
            - important properties, such as on-demand access to resources, would be maintained
            - this kind of cloud is particularly well-suited for dispersed data
    - Example: blog analysis
        - fairly dynamic data
        - blog 2 has in this example that has a bad connection
        - find a node which has a better connection to blog 2
- Summary:
    - Trends: dynamic data, mobile users, multi-cloud applications
- Discussion:
    - Using WAN emulator devices (e.g. FutureGrid, Linux kernel) for controlled experiments
    - Why only 1 proxy? With two hops figuring out paths becomes more expensive
    - Using 2 proxies in parallel? Not done yet, but will be an interesting experiment with potential benefit.
    - Martin: How to reduce the volumes of data in the future? Look first on queries on the data and move the queries as close to the sensors as you can

Steve Fisher
Data in Particle Physics

- input from Ian Bird, David Groep, Stephen Burke
- WLCG: >140 sites, ~250k CPUs, ~100 PB disks
- 6 months of data collection at LHC => ~5 PB
- Tier 0:
    - Accepts data at average of 2.6 GB/S; peaks > 7 GB
    - S
    - erves data at average of 7 GB/s
    - CERN Tier 0 moves ~1PB per day
- Jobs:
    - 1 mio jobs per day
    - >>100k CPU-days/day
    - actually much more inside pilot jobs
- Experiments:
    - Atlas and CMS with around 1000 physicists each
    - LHCb and Alice with around 200 each
    - “Experiments don’t share much code but they do learn from each other”

- Data flows:
    - four house-size detectors at CERN (the sensors)
    - most data are not recorded - multiple levels of triggering
    - results must be corrected within the simulation (because not events are recorded)
    - data reductions - using a “train”
    - data is re-analyzed 2-3 times a year
- Monarc (~2000)
    - strictly tiered approach: tier 0 (CERN) - tier 1 (national centers) - tier 2 (regional centers) - tier 3 (local)
    - data moves along the lines from tier 0 - 3
    - tier 3 never talks directly to tier 0
- Data Organization (Typical)
    - natural unit is the event (~1.5 MB)
    - events are grouped into files (~5 GB)
        - root format (home-grown c++ object persistence - grown over 10 years) - all experiments are using this format
        - central (t0) catalog has names of sites with a file (or copy)
    - files are grouped by dataset (file maybe in many datasets)
    - Local File Catalogue at T1 w/ filename to SURL mapping
    - Site service (SRM) with SURL to TURL mapping
    - no security
- Job Submission
    - WMS has problems
    - Pilot jobs as workaround to WMS
        - pilot job is asking for jobs that run well on his site
        - taking into account the location of data (affinity)
- RDBMS for experimentation data
- Evolution of data management
    - network as very reliable resource as very reliable resources that can optimize the use of storage and cpu
    - disk resources as cache
    - strict tier model not longer available
    - tiers don't matter anymore - go directly to the location of data

- Data placement observations
    - small subset of data distributed is actually used
    - data is only popular for a short time (~2 weeks)
    - data duplication: Atlas 1 PB raw data, 7 PB derived data

- Data placement
    - move towards caching of data rather than strict planned placement
    - understand a distributed system built on unreliable and asynchronous components

        - network volumes will need to scale with users

- What is most needed?
    - distributed catalogues responding to changes around the world:
        - 4 features only one is catalog
        - location of files in order to derive job placement decisions
    - ability for systems to adapt if data is not where it is expected to be
    - grid-wide home directory
    - data preservation
    - object persistence scheme is less well-suited for remote access (has been improved)
    - data-aware pilot job

Afternoon Discussion Session:

- BioSciences:
    - Parts of the complexity arises from the number of files that must be managed not the volume
    - How do you determine affinity? How do you know if you had space where to replicate?
- Quantify Work/Communication ratio in the different communities (Operations per byte processed)
- Climate domain:
    - CEDA is mainly concerned with archiving data; thus, less sensor and dynamic data
    - there are use cases with dynamic data in this domain
- LHC:
    - dynamic data?
    - How much fresh data is injected every 15 days?
        - half of the data is replaced each time
        - reconstruction is done 2-3 times per year
    - “when you move 1 PB per day you’re not dynamic” (SJ)
- Cloud Computing:
    - extension of proxy mechanism using e.g. multiple criteria
    - adaptive capabilities of proxies:
        - how to feed proxies with code
- Important infrastructure issues:
    - network
    - co-location compute/data
    - how to tie e.g. workflow components together
- Workflows: BioSciences Why Moteur?
    - designed for medical images
    - personal preferences (familiarity)
- Climate: Infrastructure for data nodes?

    - tomcat server
    - OpenDap protocol
    - transfer with GridFTP
    - each Gateway has its own registry
    - peer to peer sharing of catalogues? there is a master gateway
- BioSciences: out-of-the-box use of gLite replica service
- Jon Blower: Low Overhead distributed computing:
    - “I'd like to bring up the question of low-overhead distributed computing. Condor works really well when each job is, say, > 1 minute long. I'd like to find an approach that works when each job is a few seconds long or even much less. MapReduce seems to fit the bill at least on Google's infrastructure - but can we create a runtime system that has a similarly low overhead? What's the overhead on Hadoop?”

Day 2
Simon Dobson
Programming in data-intensive environments

Computer Science is the “third pillar” of science alongside theory and experiments - “the new microscope”

How does huge data change the game:
- Uncertainty, imprecision, timeliness
- If you deal with sensors, you have to deal with: fusion, partiality, diversity, conflict
- interoperability, longevity of data; interaction, visualization with/of data

Needs a system perspective:
- Can we trust the input?

New programming paradigm - very occasional move a problem from impossible to trivial
The Von Neumann bottleneck
- treat data in small chunks within programs
- imperative: iteration; functional: tail-recursion
- neither is efficiently parallised

Possible approaches:
- MapReduce: If you can express the problem in this way, it can be effectively parallelised and distributed
- Separate adaptation and coordination from algorithm (manually or autonomically)
    - Example for this are workflows

Garbage in (data with noise e.g.), garbage out

If the parameters don’t meet the rely conditions, the results won’t (always) meet the guarantee condition

Programming with uncertainty: “Push the error bars down in the programming language”

- any sensor-driven system has inherent uncertainty in its results
- if we use these as part of a system, wee need to track and maintain view of these errors
- use e.g. ensemble of models & compare to work around this

Migrate control around the system

- migrate control down: autonomic systems
    - add in-flight processing down within the network
    - generate code from descriptions
- migrate metadata up: semantic technology

What can you do in programming? Skeletons:

- MapReduce:
    - 2 operations that we know we can do efficiently
- Are there other equivalents?
    - e.g. sensor driven computations

Mission Languages

- Goal: capture the mission of an adaptive system

Openness

Semantic technologies:

- XML, RDF, OWL
- RDF has a strong focus on inference

How unlike programming!

- Programs that have access to their own structure and mission
- Integrate components within complex “workflows” that are exhibit adaptation
- Can be done with “normal” tools, but do we want to?

Why do we need to reassess the programming model for dynamic data?

- not all scenarios require a new programming model
- sensor networks e.g. are very different: failure and in particular partial, transient failure are the norm and quite hard to deal with
- classic programming system don’t deal with error bars and uncertainty

“QoS for data-access” as a primitive - also QoS for other things, such as some network links or file transfers? or for some modules/components? could lead to choices of replication for certain activities

Trend toward GUIs is a big obstacle for automation and for glueing apps together
middleware seems to get a bad reputation that are invented in a vacuum

Discussion:

- integrate sensing
- integrate analytics
- manage computation and workflow

Discussion Session: **Coupling data capture to simulations through to data analytics**

BioSciences:

- data must be manually partitioned for workflow steps
- Is a database the right solution for this?
    - we need file transfer
    - events must be filter (and respectively be dropped)

Fault Tolerance

- What does it mean if the workflow manager should handle exceptions? At what level do you handle exceptions in your workflow?
    - Taverna, Kepler, Moteur have basic fault tolerance
    - How do you monitor failures?
    - How do your react on failures? re-try mechanisms have their limitations
- How do you ensure that the statistics are not biased in case of partial failures
- Correctness of scientific applications is hard to define. Repeatability is a workaround for this
- Fault tolerance problem is not specific to dynamic, distributed data. It's a general challenge in distributed systems.
- All or nothing semantics: If I require 5 files to transfer, it's not worth to optimize the transfer of a single of these files.
    - Streaming algorithms that work on a partial dataset resp. the data that you have
    - Kalman Filter: Can you give a partial answer, e.g. if you only have 25/100 results?
    - What can you do in the middleware in order to address these different requirements? So that the middleware knows how many files are required.

Sensor networks:

- adaptive sampling in sensor networks triggered by certain events (e.g. a detected pollution)
- coupling the way of sensing with the model
- 2 out of 3 components: data capture & analytics

TeraGrid:

- relatively static datasets - changing at most after a couple of hours

Are there obvious and/or non-obvious application vectors for dynamic, distributed, data-intensive applications?

- DPA Vectors: coordination, communication, execution environment
- What do we need in the application vectors to incorporate dynamic data?
- What are the tunable vectors in a dynamic context? tuning by application vs. tuning the application
- Does the coordination mechanism change in a 3D application. It could, but it does not

need to. Unlikely that coordination changes, likely only certain parameters are adjusted. Dynamic data could influence how your workflow is executed. You just reconfiguring your coordination vector.

First principle approach based on dynamic data scenarios:

- BioScience:
    - no signals are processed, thus, no dynamic data
    - sequencer data is pre-recorded and not dynamic and not realtime
- Definition of dynamic?
    - Is data replication is dynamic?
- Databases:
    - out of the 1 PB data 10 TB are stored in a database (just the reference to the data)
    - DB are good for data management, meta-data, derived data etc.
    - well-suited for dynamic data
    - limited for running analyses on data, bulk data
    - “clouds are good for centralizing compute, db are good for centralizing data”
- What kind of infrastructure support?
    - partial file support on all levels

Dynamic Data Scenarios:

- Not all available but could be, versus cannot all be available
- Application Data is Dynamics versus Infrastructure Status Data (eg Monitoring)
- “If you miss it, you lose it”

3DPAS Dynamic Distributed Data-Intensive Programming Abstractions and Systems Theme -- First Workshop

What is "Dynamic Data" and why bother (Shantenu, Omer, Neil, Simon, and Dan)

-- Layout several scenarios first, and then formulate/define Dynamic Data
-- Dynamic ==> transformed by the workflow scenario (state is captured by immutable objects)
-- How does this differ from Scientific Databases
-- How does it relate to Scientific Data Management and Scientific data flow
-- Carve out the specifics of this theme -- and how it relates to such existing approaches (a diagram would be nice)
-- Underlying data model (operations) are highly domain specific -- make clear what can be domain specific and domain independent
-- Relationship to "legacy"
-- Dynamic data can arise at different levels (application vs. middleware (infrastructure)). Identify which is the primary consideration here, and what is different.
-- Reference to EU 2030 Objectives (part of the Project Europe 2030 Report) -- underpins "Data Intensive Research" from an EU perspective (strategy) + relate this to NSF "CF21" vision

How does Dynamic Data relate to Distributed & "Big" Data

-- Relate to DPA and Malcolm et al.'s theme (Data Lifecycle, Complexity, etc)
--

Application Scenario Characteristics for Dynamic Data (Various)

Identify: (1) what is the case scenario within the application; (2) what data is dynamic; (3) how the data is being used in the context of each of these scenarios:

(a) Silvia's BioSciences app: NGS, medical imaging **(Silvia)**
-- Infrastructure monitoring for the execution of workflows to manage failure -- the application would be sequence alignment (split data (this is where there is dynamic decision making) to execute the alignment in parallel)
-- DTI Imaging: two cases are interesting: DTI atlas (split, run, merge, split, merge - reduction from 10000 tasks to 1 output); PCA for classification of patients/control

(b) Climate from BADC + Oceanographic data (Data Nodes + Gateways) **(Jon Blower & Stephen Pascoe)**
-- Dynamic aspects not clear -- primarily providing a data storage (curated) system
-- Multiple sources of data -- and potentially coupling between simulation and data
-- Issue of placement and scheduling of the data -- through the use of federated registries
-- Identify how this links it to the "application genie"

(c) WLCG (focus on ATLAS) **(Steve Fisher)**

-- Most of the dynamism is in infrastructure not in the application (e.g. use of PilotJob)

-- Application dynamism: location of file (e.g. if file turns out to be not local, look up in registry-- but not do this again)

-- No specific QoS issue concerned with data transfer times

(d) AstroPhysics -- Virtual Observatory **(Bob)**

-- Analysis of data to generate event streams --> lead to the positioning of telescopes

(e) Sensor Network App (would be interesting to have) -- perhaps from **(Simon, Omer)**

-- Environmental (Marine) -- sensing within a hostile environment, data quality + availability of comms infrastructure (Simon)

-- Patient monitoring -- data quality, type of data to transmit (Omer)

Additional sources:

-- XLDB workshop series (characterising aspects of dynamic data)
http://www-conf.slac.stanford.edu/xldb10/Program.asp

-- http://www.scidb.org/

-- Digging into Data (US & UK - but somewhat global too) - http://www.diggingintodata.org/

-- Ocean Observatories Initiative (OOI, US-funded) -- Dan to talk to them

-- Additional applications from:

Manish Parashar -- Shantenu or Omer

Brian Lawrence -- Jon (Oceanographic Data)

Roger Barga (Microsoft) -- Shantenu

John Polak and Haibo Chen (Intelligent Transport) -- Omer

Bartosz “Bartek” Dobrzelecki (EPCC) -- Health Informatics and Digital Humanities -- Adam Carter

Animal tracking (Bird tracking) Willem Bouten -- Silvia

LOFAR (Low Frequency Radio Array) -- called “ASTROWISE” -- also supports provenance tracking -- Bart Scheer

TNO +Thales (Netherlands) -- Omer (Kees)

Astro Wise (Astronomical Wide-field Imaging System for Europe) http://www.astro-wise.org/

....

....

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.

Infrastructure capabilities and requirements (Shantenu, Jon, Omer, Neil, Simon, and Dan)
-- What should be provided by infrastructure (middleware, RDBMS, Cloud, Grid etc) vs. application
-- What are application characteristics: QoS-requirement vs. fault tolerance (automated) -- relate this back to the applications identified above
-- In-stream processing vs. step-wise processing of data

Diagram suggestion (perhaps indicate what is currently available and what is required (desired): -- **(Shantenu, Omer, Neil, Simon, and Dan)**

Dynamic Scenarios in Application	Vectors	Programming Systems	Infrastructure Support (Middleware ++)	Gap Analysis (What is Desired)

References:
http://ec.europa.eu/information_society/newsroom/cf/itemlongdetail.cfm?item_id=6204
http://www.nsf.gov/pubs/2010/nsf10015/nsf10015.pdf

Expectations for theme:
- Definition of dynamic data
- Survey-like paper

Discussion with Malcolm
Is dynamic data the wrong term for branding?
REBRAND DYNAMIC?! Call it static?

HPDC workshop:
https://sites.google.com/site/3dapas/