Combining Data-Intensive with Modelling:
to make the most of data and computation

Malcolm Atkinson

Malcolm.Atkinson@ed.ac.uk

28th May 2012

1st EPOS-Orfeus Coordination Meeting

Global challenges for seismological data analysis

EMFCSC, Erice

FP7-INFRASTRUCTURES-2011 project# 283543

Outline

picture from Erika Salmon
Cornish Coast Path
where I call home

- Data Intensive
  - What is it?
  - Why use it?
- HPC & Data Intensive
  - Similarities and Differences
- Models for Coupling
  - Loose coupling
  - Examples
  - Tight coupling
- Sketch of Data-Intensive thinking
- Summary and Conclusions

Data-Intensive Thinking

Gray's Laws of Data Engineering

Jim Gray:

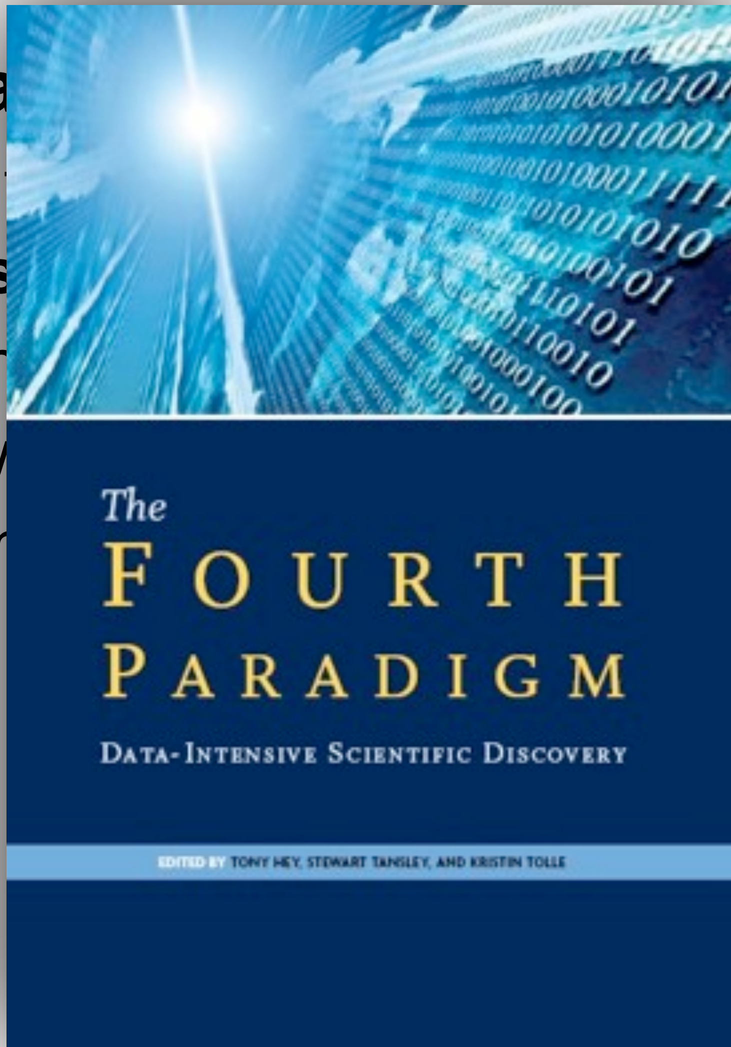
- Scientific computing is revolving around **data**
- Need **scale-out** solution for analysis
- Take the **analysis to the data**!
- Start with "**20 queries**"
- Go from "**working to working**"

From: Alex Szalay, JHU

Gray's Laws of Data Engineering

Jim Gra

- Scienti … around **data**
- Need **s** … ysis
- Take th
- Start w
- Go fro



From: Alex Szalay, JHU

Defining “Data-Intensive”

Defining "Data-Intensive"

- **Generally**
  - ***A computational task is data-intensive if you have to think hard about an aspect of data handling to make progress***
    - distribution, permissions and rules of use, complexity, heterogeneity, rate of arrival, unstructured or changing structure, long tail of small and scattered instances, size of data, number of users
    - often in combination

Defining "Data-Intensive"

- **Generally**
  - ***A computational task is data-intensive if you have to think hard about an aspect of data handling to make progress***
    - distribution, permissions and rules of use, complexity, heterogeneity, rate of arrival, unstructured or changing structure, long tail of small and scattered instances, size of data, number of users
    - often in combination
- **Quantitatively**
  - **The computation's Amdahl numbers are close to 1**
    - CPU operations : bits transferred in or out of memory
    - 1000 CPU operations : 1 I/O operation
  - **Total volumes expensive to store**
  - **Total requests/unit time hard to accommodate**

Data-Intensive Strategies 1

Data-Intensive Strategies 1

- Use commodity components and low power
  - So that you can afford a lot of them
  - Balanced for data-intensive work
  - Treat memory bandwidth as a scarce resource

Data-Intensive Strategies 1

- Use commodity components and low power
  - So that you can afford a lot of them
  - Balanced for data-intensive work
  - Treat memory bandwidth as a scarce resource
- Data & computation as close together as possible
  - in the processor cache in fewest steps & not disrupted

Data-Intensive Strategies 1

- Use commodity components and low power
  - So that you can afford a lot of them
  - Balanced for data-intensive work
  - Treat memory bandwidth as a scarce resource
- Data & computation as close together as possible
  - in the processor cache in fewest steps & not disrupted
- Work on small chunks of data
  - as small as logically possible
  - a column of a table
  - a row of a table
  - a file
  - data unbundled, in computational format & compressed

Data-Intensive Strategies 1

- Use commodity components and low power
  - So that you can afford a lot of them
  - Balanced for data-intensive work
  - Treat memory bandwidth as a scarce resource
- Data & computation as close together as possible
  - in the processor cache in fewest steps & not disrupted
- Work on small chunks of data
  - as small as logically possible
  - a column of a table
  - a row of a table
  - a file
  - data unbundled, in computational format & compressed
- Once data is close to a processor do all you can with it
  - multiple derivatives in one pass
  - pipelining
  - re-use of intermediate data, caching and forwarding

Data-Intensive Strategies 2

Data-Intensive Strategies 2

- **Exploit very large scale parallelism and distribution**
  - many subtasks at modest rate per task in large numbers
  - NOT tightly coupled parallelism!!!
  - distribution for availability, ownership & persistence
  - proximity to data sources or destinations for speed

Data-Intensive Strategies 2

- **Exploit very large scale parallelism and distribution**
  - many subtasks at modest rate per task in large numbers
  - NOT tightly coupled parallelism!!!
  - distribution for availability, ownership & persistence
  - proximity to data sources or destinations for speed
- **Replicate**
  - for more parallelism and for durable persistence

Data-Intensive Strategies 2

- **Exploit very large scale parallelism and distribution**
  - many subtasks at modest rate per task in large numbers
  - NOT tightly coupled parallelism!!!
  - distribution for availability, ownership & persistence
  - proximity to data sources or destinations for speed
- **Replicate**
  - for more parallelism and for durable persistence
- **Most data WORM (Write Once Read Many)**
  - or WORN (Write Once Read Never) - automatically eliminate or clean up

Data-Intensive Strategies 2

- **Exploit very large scale parallelism and distribution**
  - many subtasks at modest rate per task in large numbers
  - NOT tightly coupled parallelism!!!
  - distribution for availability, ownership & persistence
  - proximity to data sources or destinations for speed
- **Replicate**
  - for more parallelism and for durable persistence
- **Most data WORM (Write Once Read Many)**
  - or WORN (Write Once Read Never) - automatically eliminate or clean up
- **Updates local and mostly append (mostly non-Transactional)**

Data-Intensive Strategies 2

- Exploit very large scale parallelism and distribution
  - many subtasks at modest rate per task in large numbers
  - NOT tightly coupled parallelism!!!
  - distribution for availability, ownership & persistence
  - proximity to data sources or destinations for speed
- Replicate
  - for more parallelism and for durable persistence
- Most data WORM (Write Once Read Many)
  - or WORN (Write Once Read Never) - automatically eliminate or clean up
- Updates local and mostly append (mostly non-Transactional)
- Coordination & Catalogue DBs
  - distributed shared structures
  - just enough synchronisation

Data-Intensive Strategies 2

- Exploit very large scale parallelism and distribution
  - many subtasks at modest rate per task in large numbers
  - NOT tightly coupled parallelism!!!
  - distribution for availability, ownership & persistence
  - proximity to data sources or destinations for speed
- Replicate
  - for more parallelism and for durable persistence
- Most data WORM (Write Once Read Many)
  - or WORN (Write Once Read Never) - automatically eliminate or clean up
- Updates local and mostly append (mostly non-Transactional)
- Coordination & Catalogue DBs
  - distributed shared structures
  - just enough synchronisation
- Fine-grained local protection & authorisation

Data-Intensive Strategies 2

- Exploit very large scale parallelism and distribution
  - many subtasks at modest rate per task in large numbers
  - NOT tightly coupled parallelism!!!
  - distribution for availability, ownership & persistence
  - proximity to data sources or destinations for speed
- Replicate
  - for more parallelism and for durable persistence
- Most data WORM (Write Once Read Many)
  - or WORN (Write Once Read Never) - automatically eliminate or clean up
- Updates local and mostly append (mostly non-Transactional)
- Coordination & Catalogue DBs
  - distributed shared structures
  - just enough synchronisation
- Fine-grained local protection & authorisation
- Statistical and quantised accounting

Data-Intensive Strategies 3

Data-Intensive Strategies 3

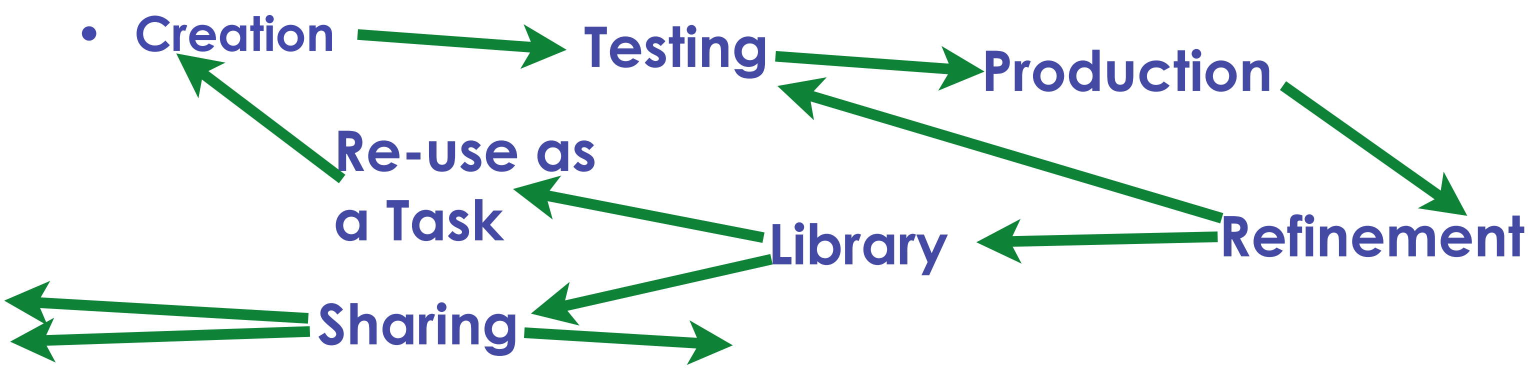
- **High-level notations for describing methods /composing tasks**
  - *with well-developed optimised transformations before execution*
  - query languages: SQL/AQL, (Xquery &SPARQL), ...
  - workflow languages: Kepler, Pegasus, DISPEL, ...
  - MapReduce: PigLatin, ZigZag, ...

Data-Intensive Strategies 3

- High-level notations for describing methods /composing tasks
    - *with well-developed optimised transformations before execution*
    - query languages: SQL/AQL, (Xquery &SPARQL), ...
    - workflow languages: Kepler, Pegasus, DISPEL, ...
    - MapReduce: PigLatin, ZigZag, ...
- Providers + Community + User definition of (libraries of) tasks
    - your signal processing, geophysics & data-presentation steps
    - your existing code & preferred languages

Data-Intensive Strategies 3

- **High-level notations for describing methods /composing tasks**
  - *with well-developed optimised transformations before execution*
  - query languages: SQL/AQL, (Xquery &SPARQL), ...
  - workflow languages: Kepler, Pegasus, DISPEL, ...
  - MapReduce: PigLatin, ZigZag, ...
- **Providers + Community + User definition of (libraries of) tasks**
  - your signal processing, geophysics & data-presentation steps
  - your existing code & preferred languages
- **Support for the query & workflow lifetime: new research objects**



Tradeoffs Today

“Extreme computing is about tradeoffs”

Stu Feldman (Google)

Ordered priorities for data-intensive scientific computing

1. Total storage (-> low redundancy)
2. Cost (-> total cost vs price of raw disks)
3. Sequential IO (-> locally attached disks, fast ctrl)
4. Fast stream processing (->GPUs inside server)
5. Low power (-> slow normal CPUs, lots of disks)

The order will be different in a few years...and scalability may appear as well

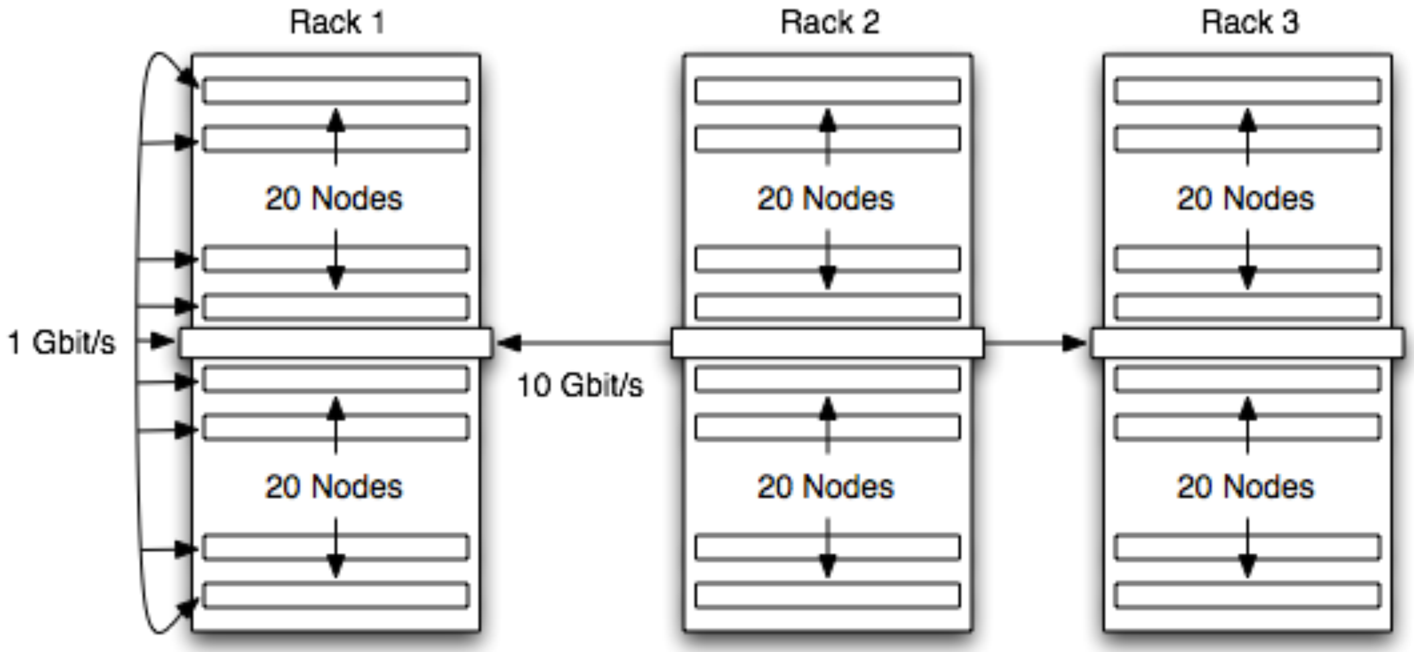
G. Bell, J. Gray, and A. S. Szalay, “Petascale computational systems: balanced cyberinfrastructure in a data-centric world,” IEEE Computer, vol. 39, no. 1, pp. 110–12, 2006.

A. S. Szalay, G. C. Bell, H. H. Huang, A. Terzis, and A. White, “Low-Power Amdahl-Balanced Blades for Data Intensive Computing,” ACM Operating Systems Review, 2010.

A. S. Szalay, “Extreme data-intensive scientific computing,” Computing in Science and Engineering, vol. 13, no. 6, pp. 34–41, 2011.

From: Alex Szalay, JHU

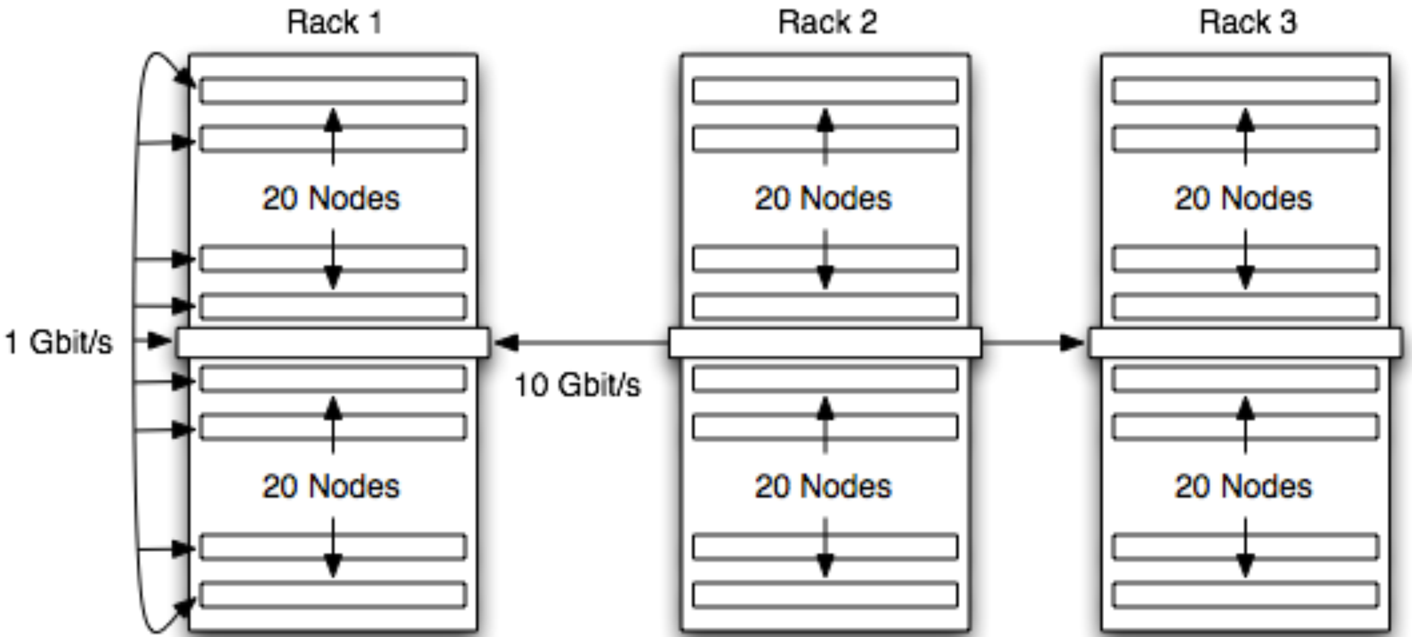
EDIM1: Our Data-Intensive Experimental Platform



Node

Motherboard	Zotac ION-ITX-K 1.6 GHz Atom/ION Mini-ITX
Processor	Dual-Core Intel 1.6 GHz Atom
Memory	4 GB DDR3
Storage	1 x 256 MB SSD (RealSSD C300)
	3 x 2 TB HDD (Hitachi Deskstar 7K3000)
GPU	NVIDIA Ion

EDIM1: Our Data-Intensive Experimental Platform



A data brick

Node

Motherboard	Zotac ION-ITX-K 1.6 GHz Atom/ION Mini-ITX
Processor	Dual-Core Intel 1.6 GHz Atom
Memory	4 GB DDR3
Storage	1 x 256 MB SSD (RealSSD C300)
	3 x 2 TB HDD (Hitachi Deskstar 7K3000)
GPU	NVIDIA Ion

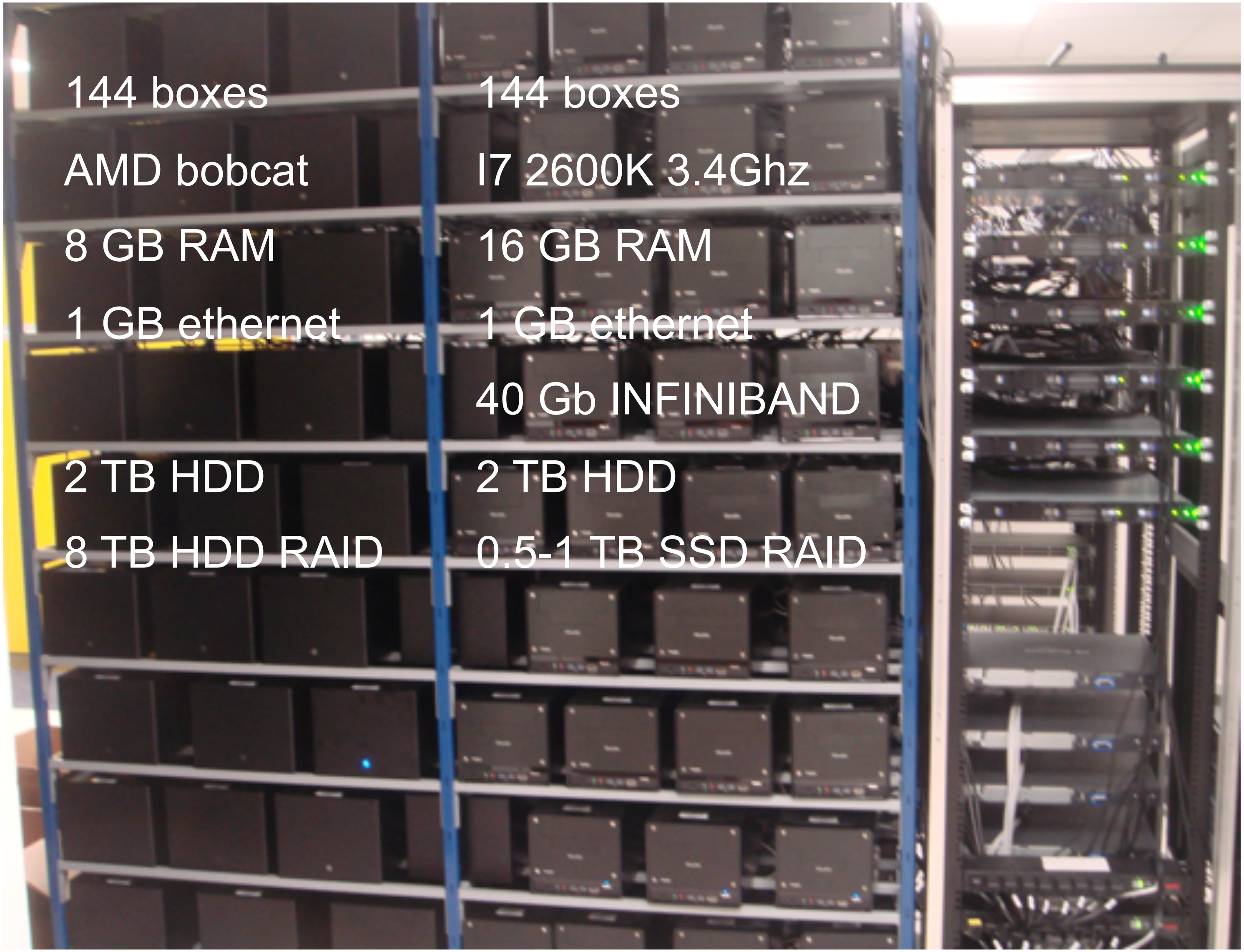
JHU Data-Scope

- Funded by NSF MRI to build a new ‘instrument’ to look at data
- Goal: 102 servers for $1M + about $200K switches+racks
- Two-tier: performance (P) and storage (S)
- Large (5PB) + cheap + fast (400+GBps), but …
  . ..a special purpose instrument

	1P	*1S*	*90P*	*12S*	*Full*	
servers	1	1	90	12	102	
rack units	4	12	360	144	504	
capacity	24	252	2160	3024	5184	TB
price	8.5	22.8	766	274	1040	$K
power	1	1.9	94	23	116	kW
GPU	3	0	270	0	270	TF
seq IO	4.6	3.8	414	45	459	GBps
netwk bw	10	20	900	240	1140	Gbps

A. Szalay, JHU

The Scilens Cluster



M. L. Kersten and S. Manegold, “Revolutionary database technology for data intensive research,” ERCIM News, vol. 2012, no. 89, 2012.

From: Martin Kersten, CWI

Data-Intensive Hardware Platform Choices

Data-Intensive Hardware Platform Choices

- Add data bricks in balance with latest technology
  - as you need them

Data-Intensive Hardware Platform Choices

- Add data bricks in balance with latest technology
  - as you need them
- But you don't want
  - to buy a brick at time
  - to manage bricks individually

Data-Intensive Hardware Platform Choices

- Add data bricks in balance with latest technology
  - as you need them
- But you don't want
  - to buy a brick at time
  - to manage bricks individually
- Buy/rent crates of data bricks
  - one to a small number per data condominium
  - ownership and management at data condo granularity

Data-Intensive Hardware Platform Choices

- Add data bricks in balance with latest technology
  - as you need them
- But you don't want
  - to buy a brick at time
  - to manage bricks individually
- Buy/rent crates of data bricks
  - one to a small number per data condominium
  - ownership and management at data condo granularity
- Federation of data condominiums
  - to match reality of ownership and control politics
  - to support geographically dispersed concentrations of work
  - to achieve availability, durable persistence, ...

Data-Intensive Hardware Platform Choices

- Add data bricks in balance with latest technology
  - as you need them
- But you don't want
  - to buy a brick at time
  - to manage bricks individually
- Buy/rent crates of data bricks
  - one to a small number per data condominium
  - ownership and management at data condo granularity
- Federation of data condominiums
  - to match reality of ownership and control politics
  - to support geographically dispersed concentrations of work
  - to achieve availability, durable persistence, ...
- Inevitable heterogeneous
  - optimum choice varies with locality and time
  - specialised architectures for specific tasks

Data-Intensive Software Choices

- Build on optimised subsystems with critical commitment behind them
- File systems
  - GFS, HFS, Sector, ...
  - S. Ghemawat, H. Gobioff, and S.-T. Leung, “The google file system,” in SOSP, pp. 29–43, 2003.
  - J. Shafer, S. Rixner, A. L. Cox: The Hadoop distributed filesystem: Balancing portability and performance. ISPASS 2010: 122-133
  - Gu and R. L. Grossman, “Sector: A high performance wide area community data storage and sharing system,” Future Generation Comp. Syst., vol. 26, no. 5, pp. 720–728, 2010
  - B. Trushkowsky, P. Bod ́ık, A. Fox, M. J. Franklin, M. I. Jordan, and D. A. Patterson, “The SCADS Director: Scaling a Distributed Storage System Under Stringent Performance Requirements,” in Proceedings of FAST ’11: Conference on File and Storage Technologies, USENIX, 2011
  - S. Patil and G. Gibson, “Scale and concurrency of giga+: file system directories with millions of files,” in Proceedings of the 9th USENIX conference on File and Storage technologies, FAST’11, (Berkeley, CA, USA), pp. 13–13, USENIX Association, 2011
  - ...

Data-Intensive Software Choices

- Build on optimised subsystems with critical commitment behind them
- Data-base systems
  - SciDB www.scidb.org
  - MonetDB www.monetdb.org
  - Microsoft SQL server
  - XLDB workshop series www.xldb.org
  - MapReduce, Hadoop, ...
  - J. Dean and S. Ghemawat, “MapReduce: simplified data processing on large clusters,” CACM, vol. 51, no. 1, pp. 107–113, 2008
  - T. White, Hadoop: The Definitive Guide. O’Reilly, 2009
  - M. Stonebraker, D. Abadi, D. J. DeWitt, S. Madden, E. Paulson, A. Pavlo, and A. Rasin, “MapReduce and parallel DBMSs: friends or foes?,” Communications of the ACM, vol. 53, no. 1, pp. 64–71, 2010

Directly mapped scientific data structures

Data-Intensive Software Choices

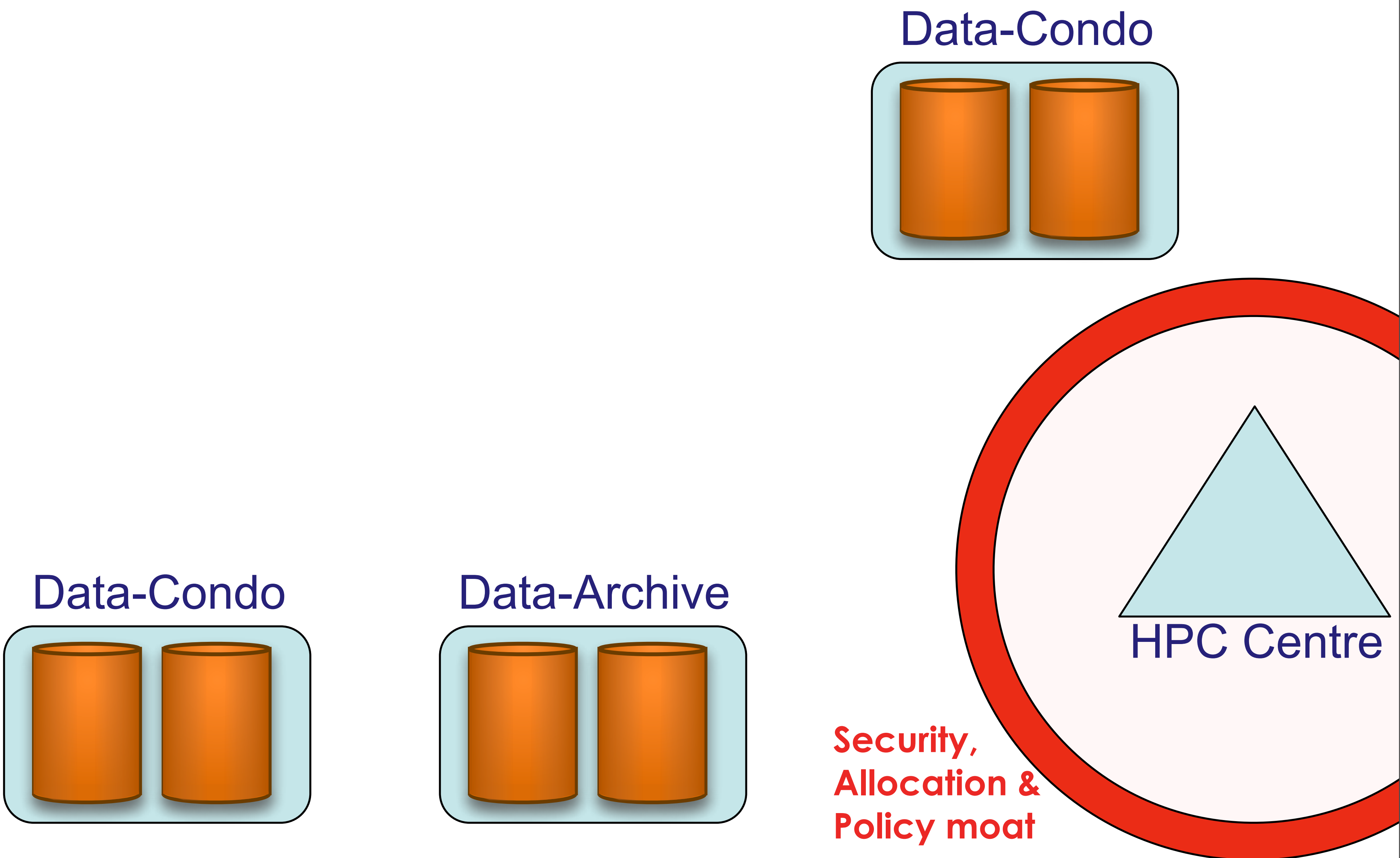
- Build on optimised subsystems with critical commitment behind them
- Shared-structured updatable memory
  - BigTable — *research.google.com/archive/bigtable.html*
  - Cassandra — *cassandra.apache.org*
  - ...
- Reliable message systems for orchestration, monitoring and control
  - rabbitmq — *www.rabbitmq.com*
  - storm-MQ — *stormmq.com*
- Data-transport systems

Data-Intensive Software Choices

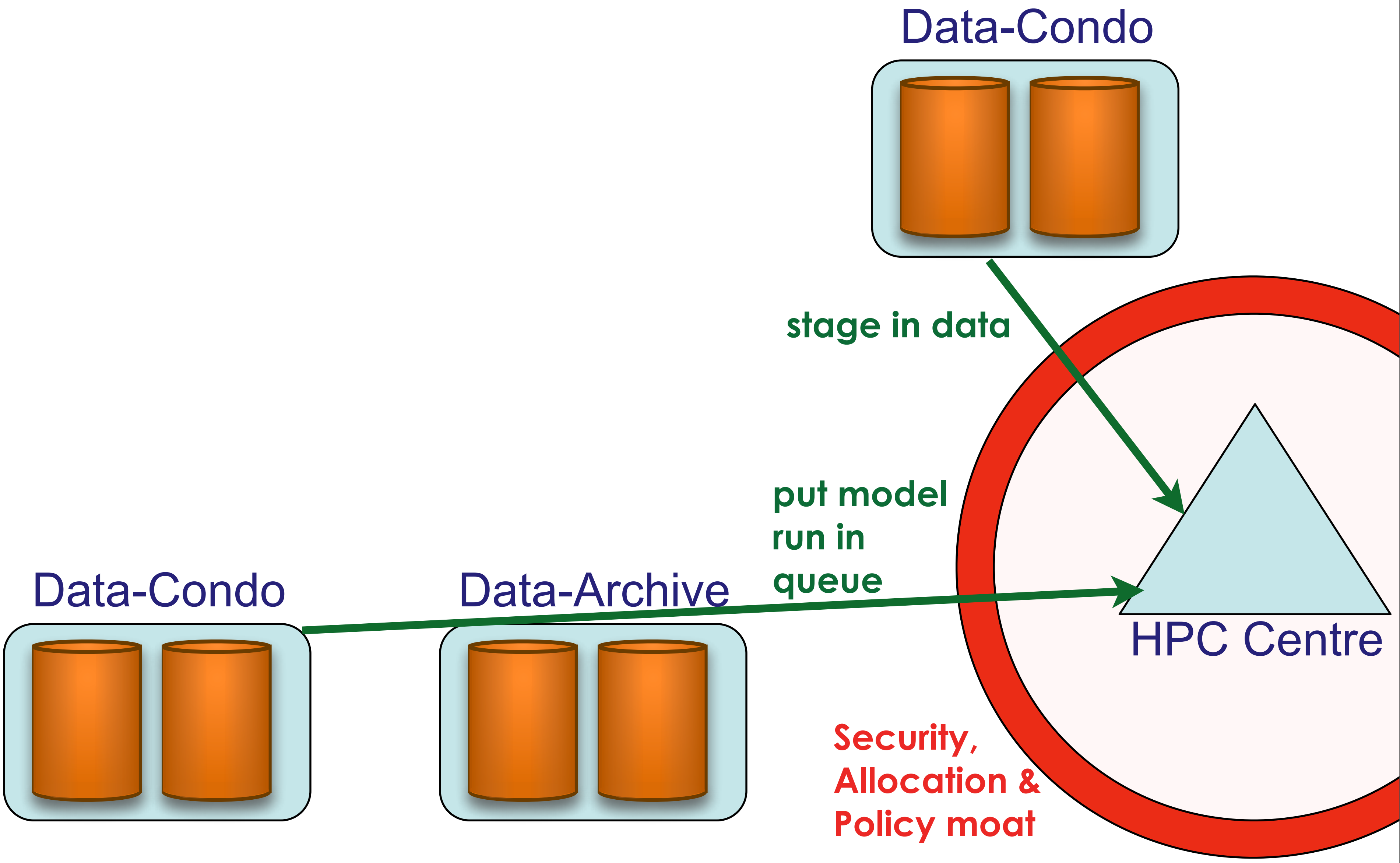
- Build on optimised subsystems with critical commitment behind them
- Data-transport systems
- Optimising workflow systems
  - Multiple scales of activity & data granularity
  - Multiple libraries of activities
  - Choices in model, editor & notation:
  - Kepler, Trident, Knime, DISPEL, ...
- Parallel execution frameworks
- Well developed libraries of components

Data-Intensive and HPC (in)compatibility

Loosely coupled model submit

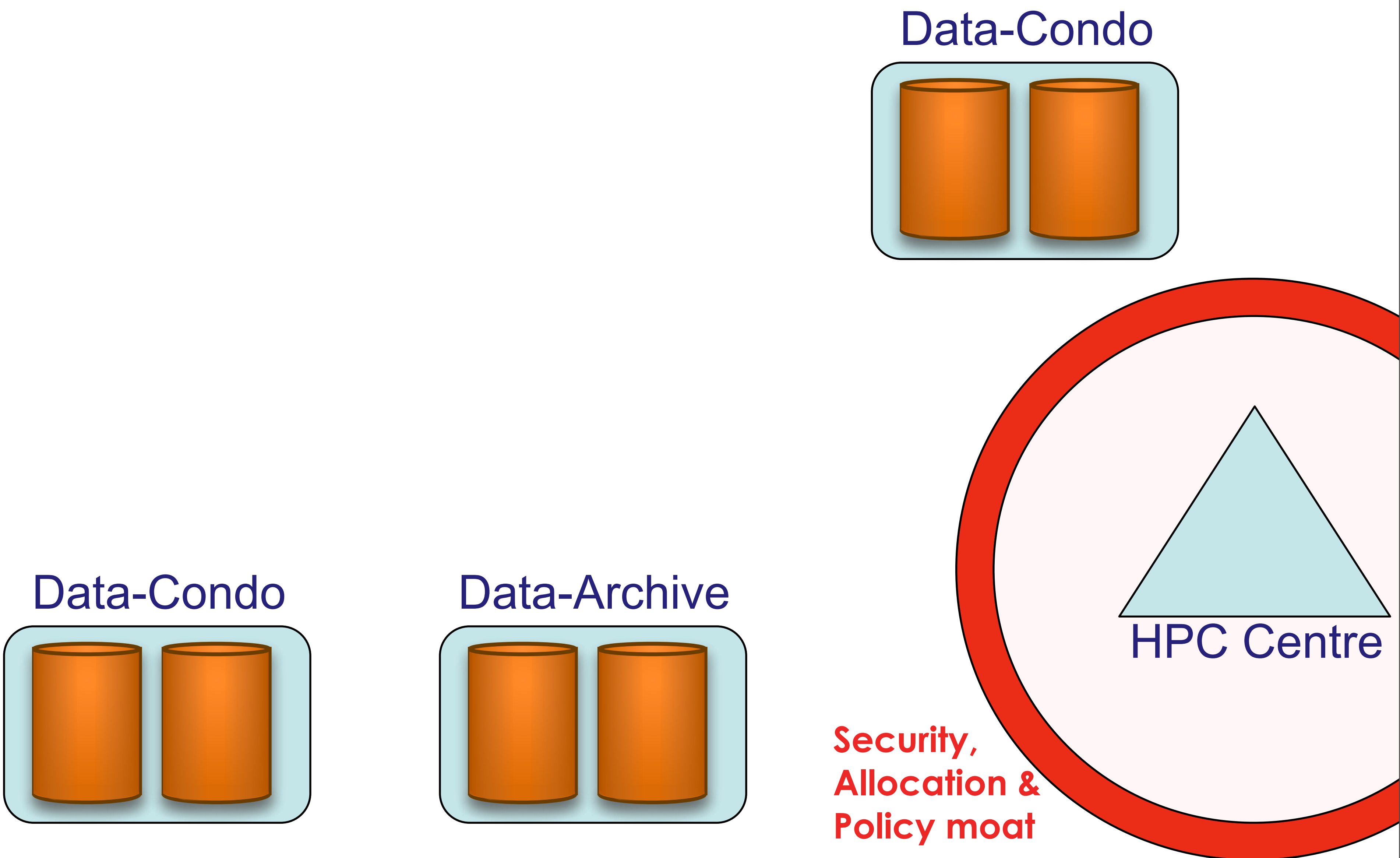


Loosely coupled model submit

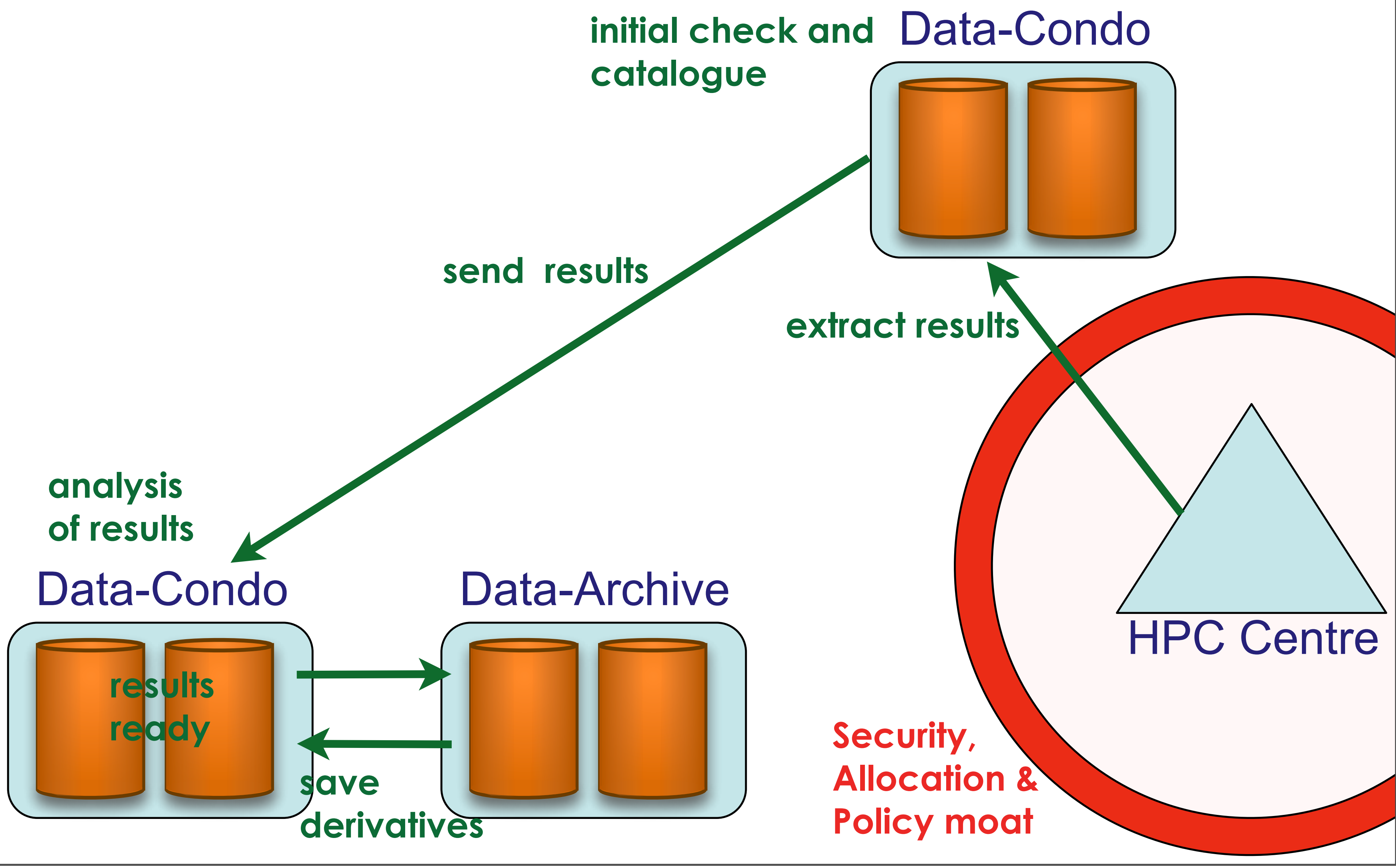


Time passes

Loosely coupled model



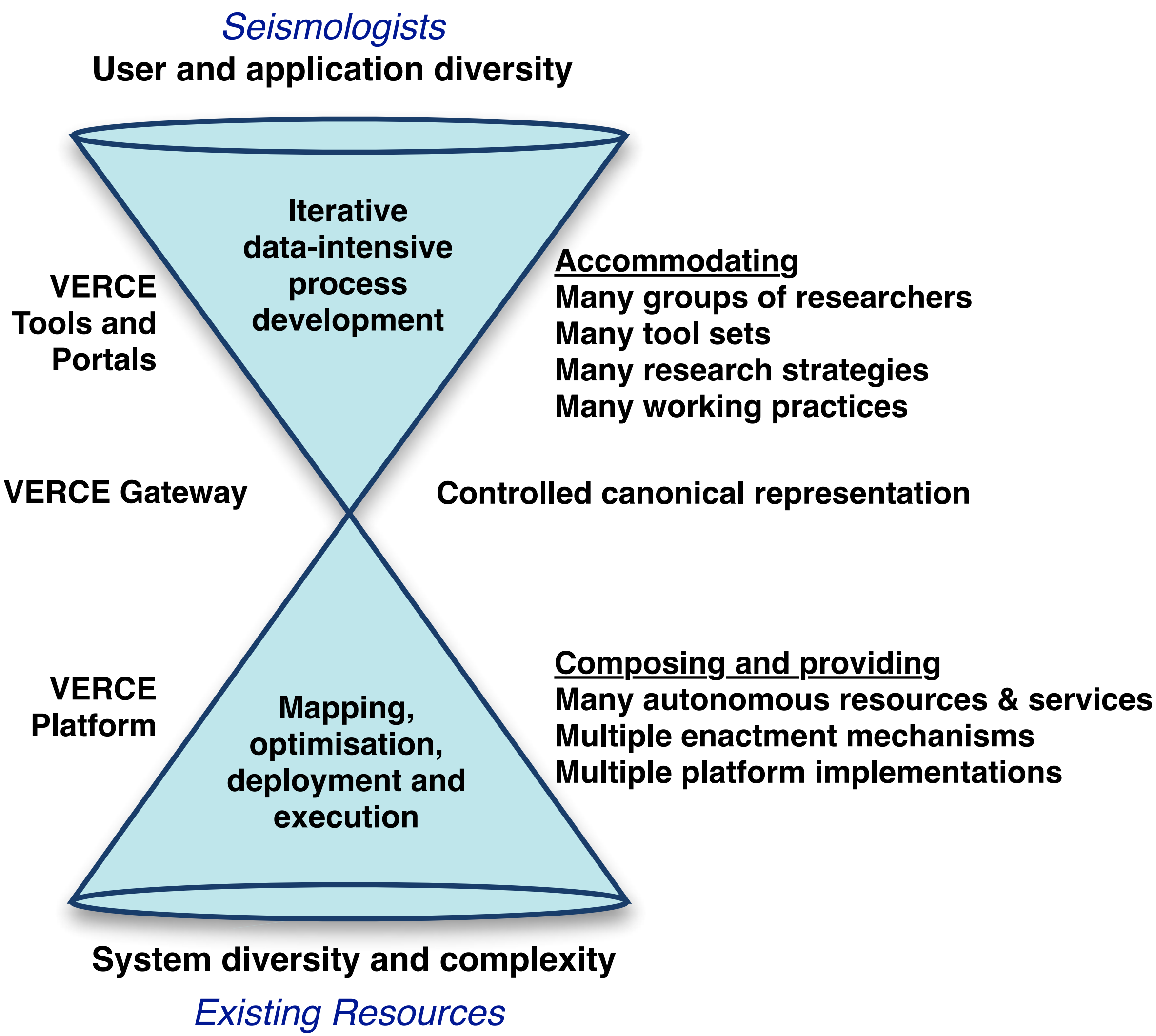
Loosely coupled model



Caveats with loose coupling

- **In principle all of the pre and post phases + the modelling job can be described in one workflow**
  - **It could be automatically partitioned**
  - **Then sent to the correct places**
  - **Data movement & Job submission can be tasks**
  - **The fragments could be automatically orchestrated**
    - E.G. the preparation workflows could pause waiting for results
    - The pause could be released when a message gets sent from job
    - The post-processing workflow fragments would then run
- **But**
  - **Today researchers have to negotiate resources, get authorisation in different regimes, get data over moat, ...**

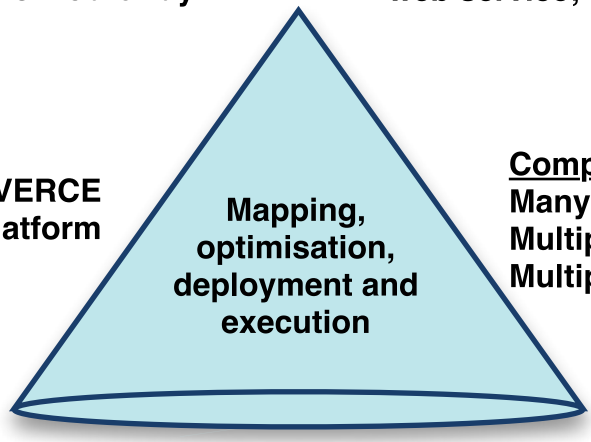
Mapping a complete request



Focus on the gateway and platform

VERCE Gateway **web service; a workflow as "query"**

VERCE Platform



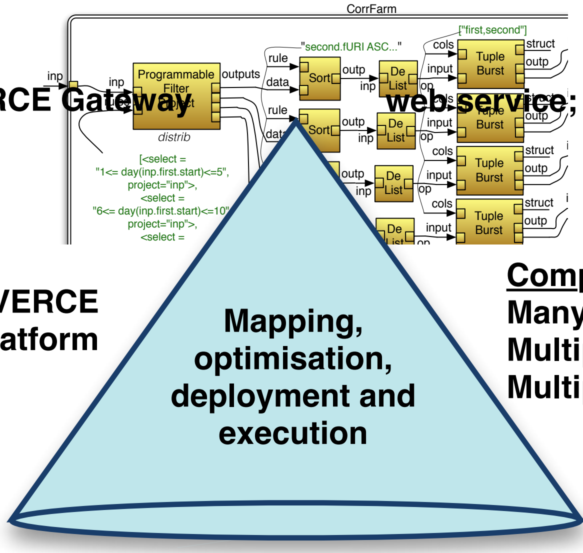
<u>Composing and providing</u>
Many autonomous resources & services
Multiple enactment mechanisms
Multiple platform implementations

System diversity and complexity

Archives, Data condominiums, Grids & HPC centres

Focus on the gateway and platform

VERCE Gateway **web service; a workflow as "query"**



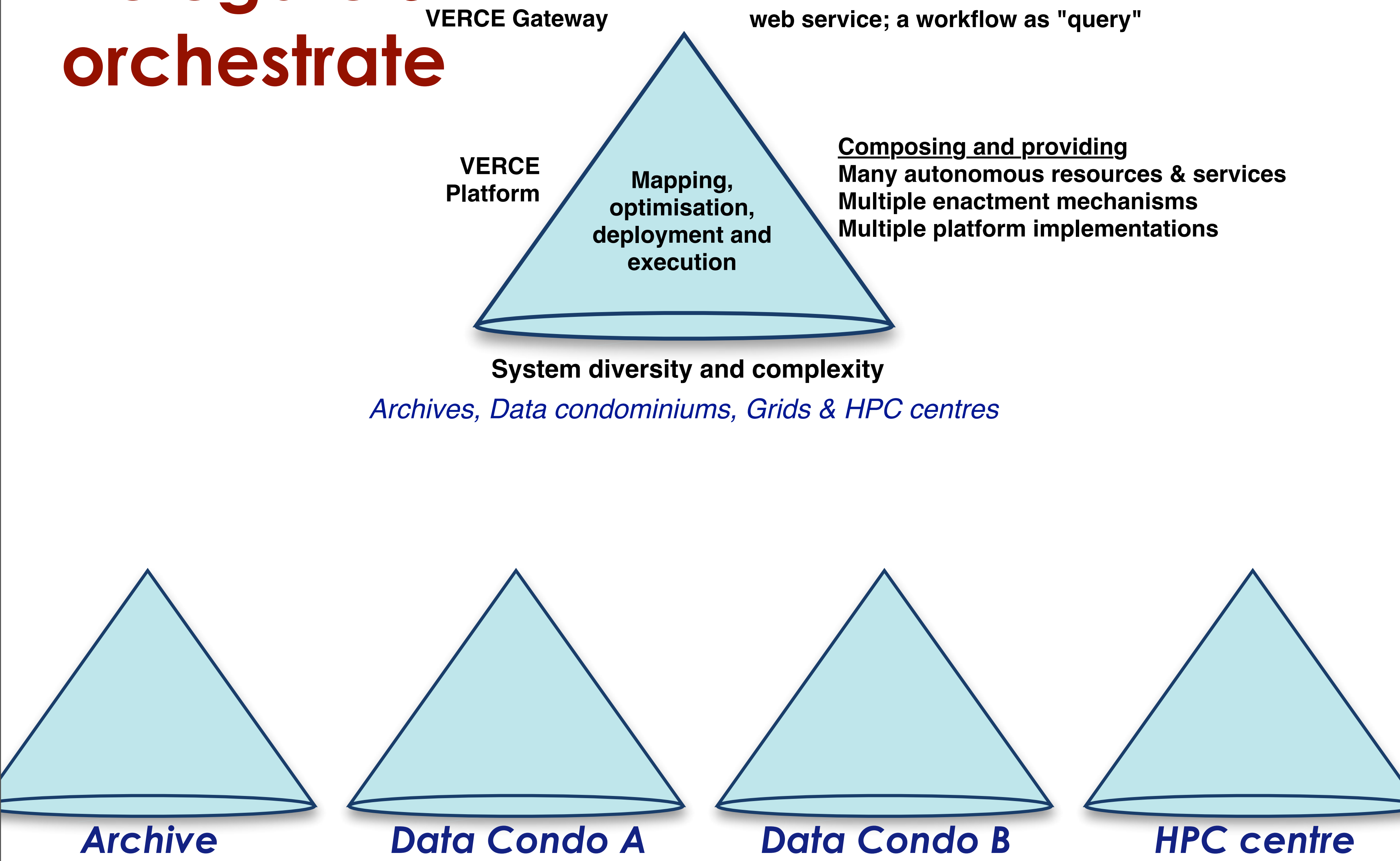
VERCE Platform

<u>Composing and providing</u>
Many autonomous resources & services
Multiple enactment mechanisms
Multiple platform implementations

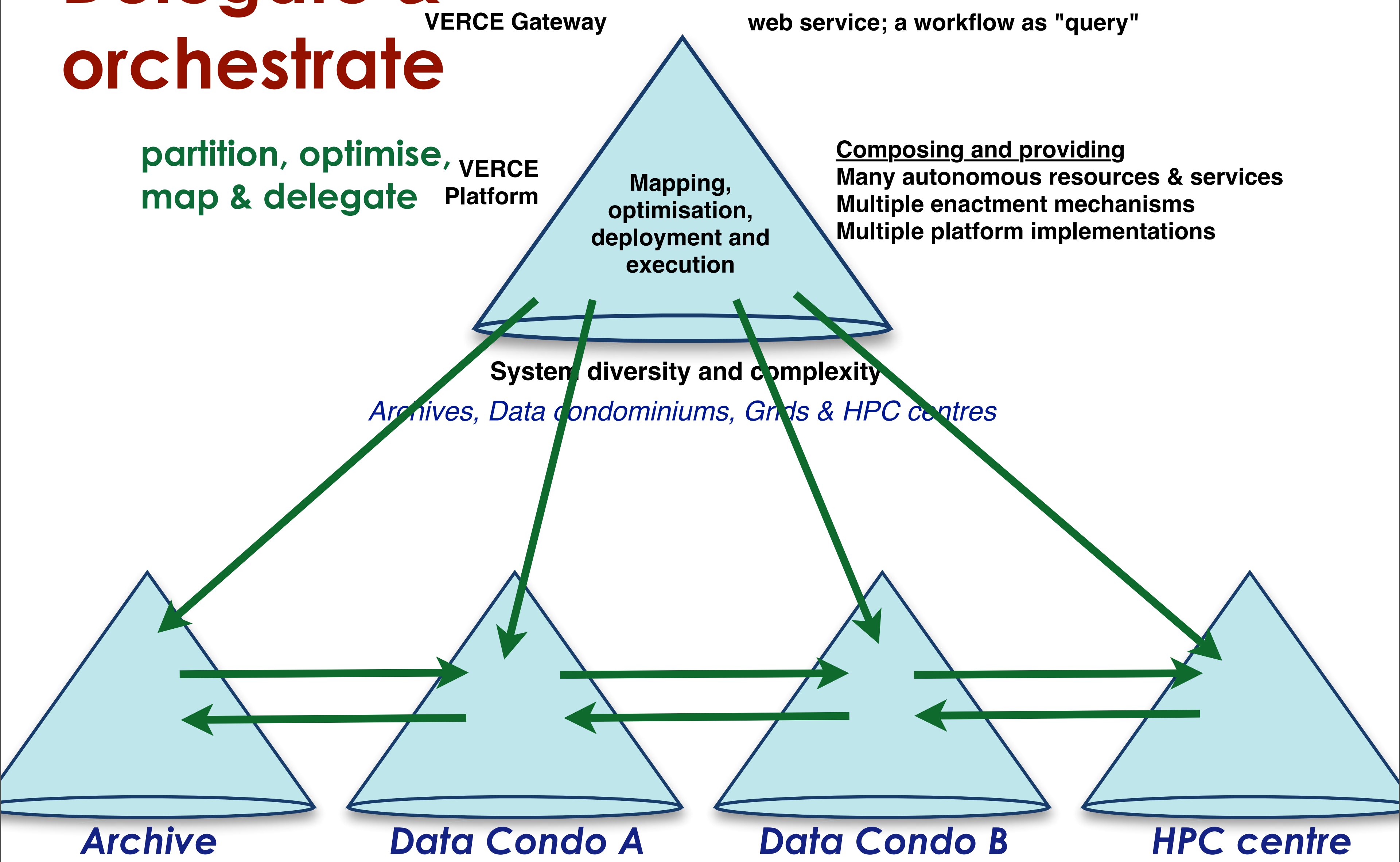
System diversity and complexity

Archives, Data condominiums, Grids & HPC centres

Delegate & orchestrate



Delegate & orchestrate



Example Data-Intensive analysis after model run

Data in HPC Simulations

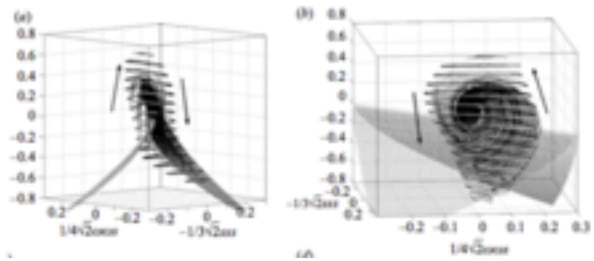
- HPC is an instrument in its own right
- Largest simulations approach petabytes
  - from supernovae to turbulence, biology and brain modeling
- Need public access to the best and latest through interactive numerical laboratories
- Creates new challenges in
  - how to move the petabytes of data (high speed networking)
  - How to look at it (render on top of the data, drive remotely)
  - How to interface (virtual sensors, immersive analysis)
  - How to analyze (algorithms, scalable analytics)

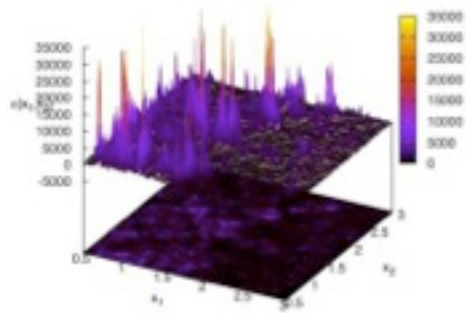
From: Alex Szalay, JHU

Immersive Turbulence

"… the last unsolved problem of classical physics…" Feynman

- **Understand the nature of turbulence**
  - Consecutive snapshots of a large simulation of turbulence: now 30 Terabytes
  - Treat it as an experiment, **play** with the database!
  - **Shoot test particles** (sensors) from your laptop into the simulation, like in the movie Twister
  - Next: 70TB MHD simulation
- **New paradigm** for analyzing simulations!





C. Meneveau (Mech. E), G. Eyink (Applied Math), R. Burns (CS), A. Szalay (P&A)
From: Alex Szalay, JHU

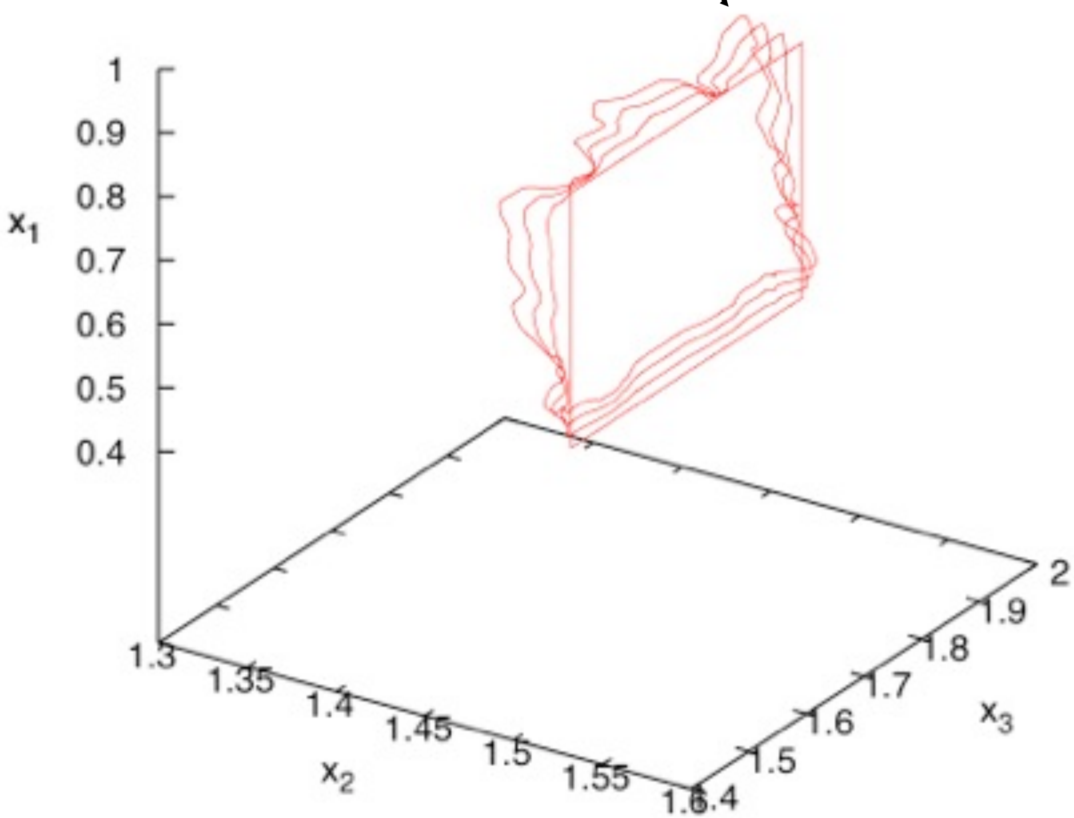
Sample code (fortran 90)

```
!
 do it = 1,15,1
!
   print *,'time = ',time
   time=time+deltat
!
   CALL getvelocity(authkey, dataset1, time, Lagrangian6thOrder , PCHIPInterpolation, 4*n, points, dataout)
 !
  do i=1,4*n
  do k=1,3
    points(k,i)=points(k,i)+dataout(k,i)*deltat
  end do
  end do
!
  if (it.eq.5.or.it.eq.10.or.it.eq.15) then
   do i=1,4*n
    write(10,*) points(1,i),points(2,i),points(3,i
   end do
    write(10,*) points(1,1),points(2,1),points(3,1
  endif
  write(10,*) '  '
  end do
!
  endif
```

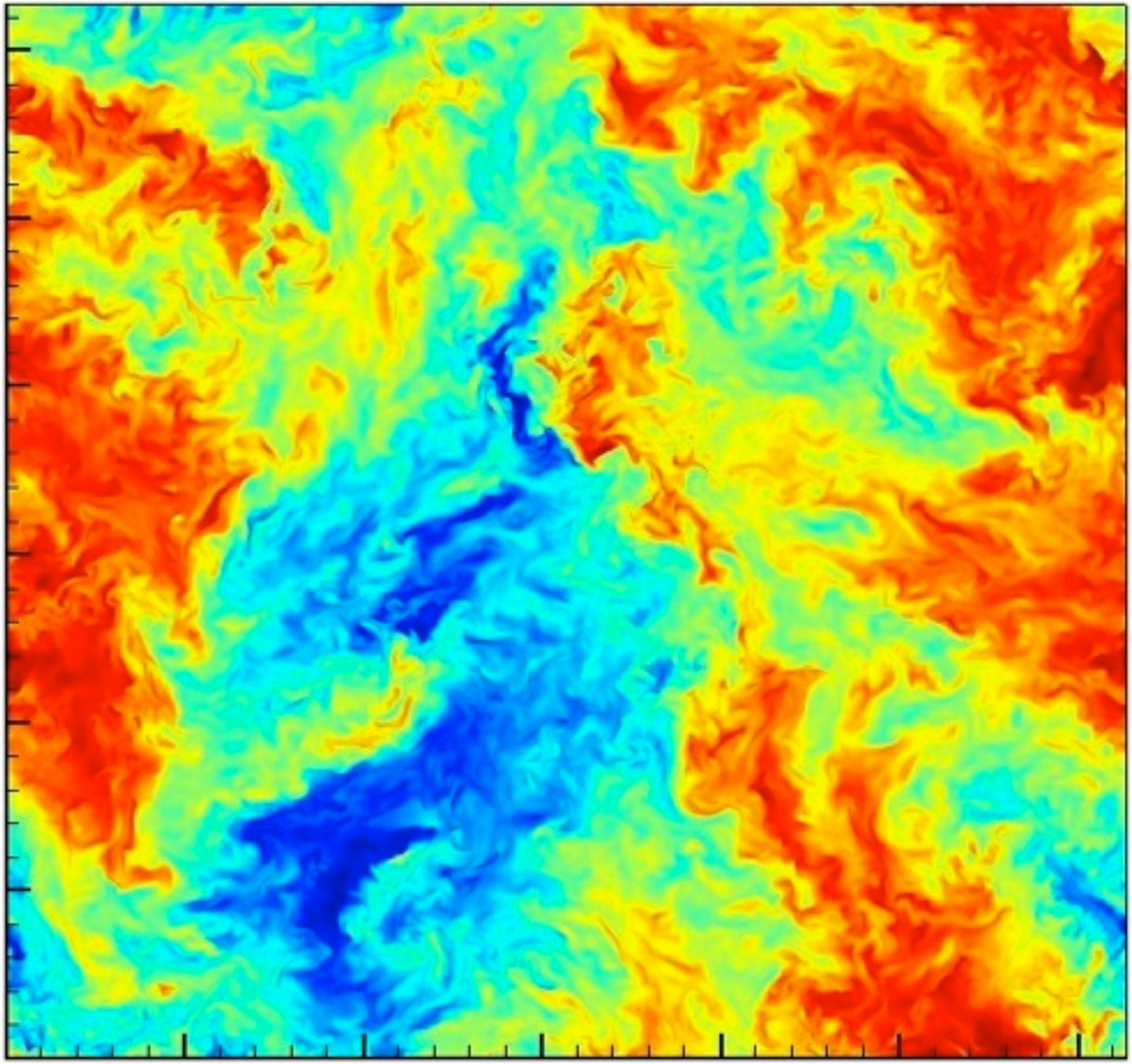
minus

advect backwards in time !

Not possible during DNS

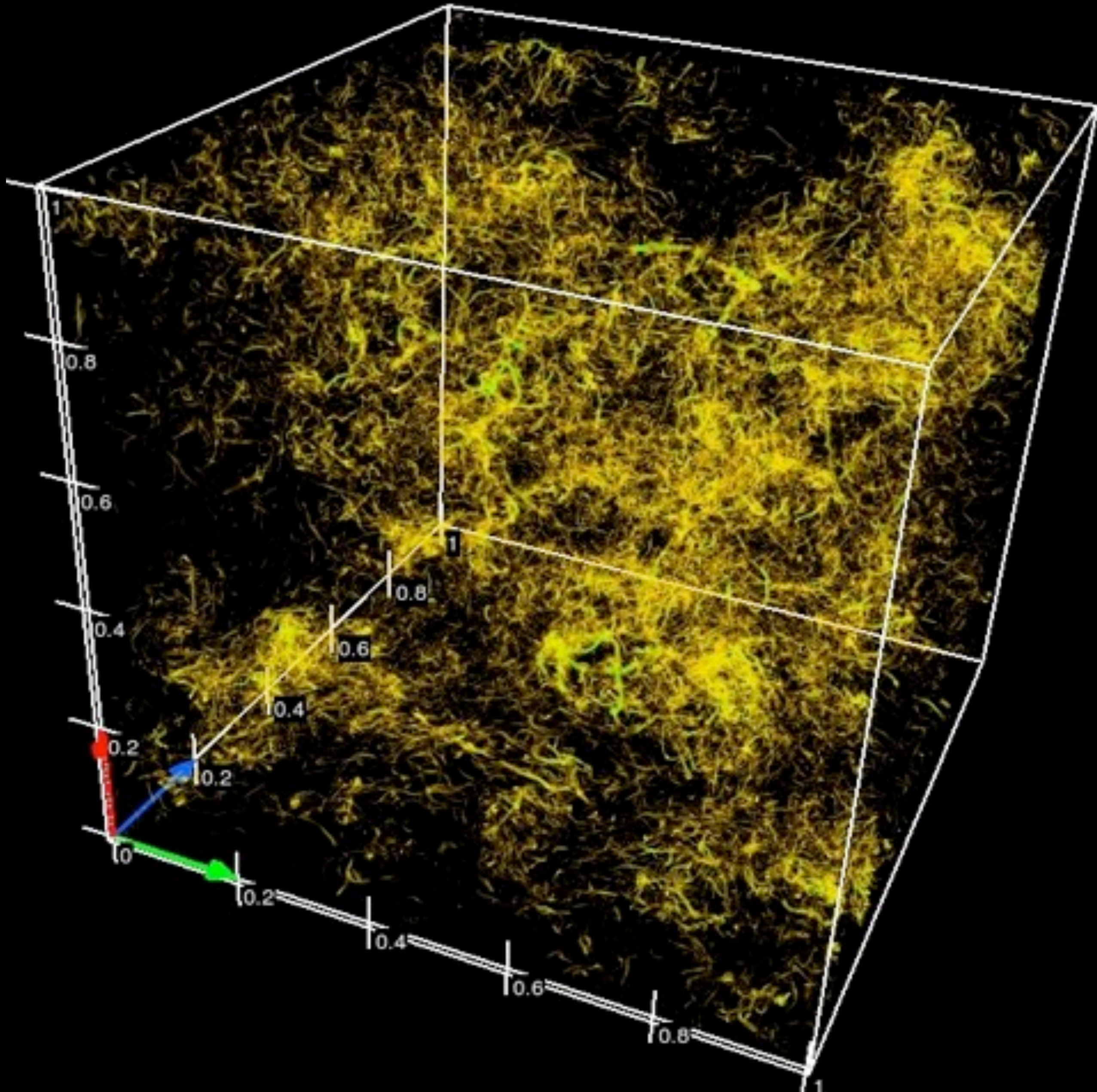


From: Alex Szalay, JHU



$u(x,y,z_0,t_0)$ extracted from database using Matlab (C. Verhulst)

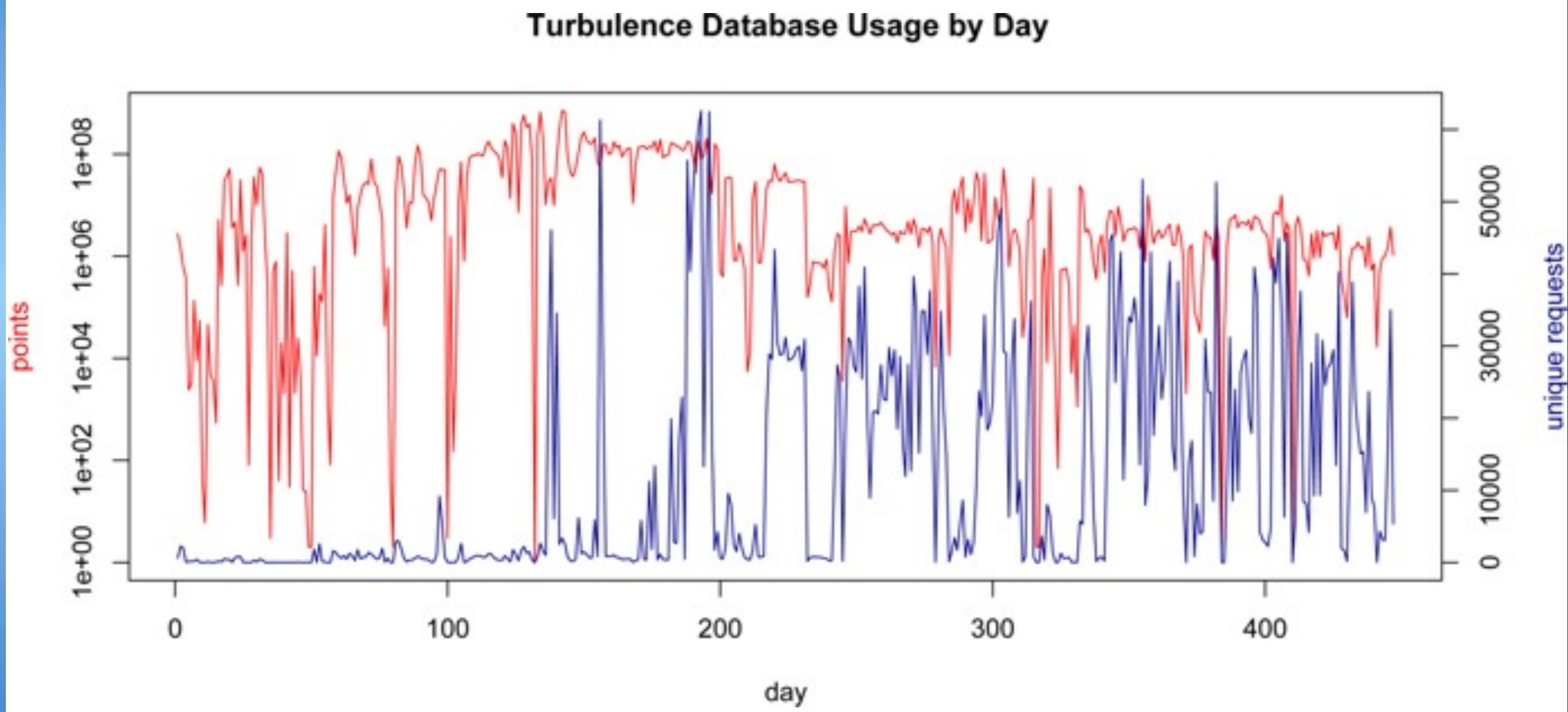
From: Alex Szalay, JHU



Vorticity magnitude extracted from database
using C (J. Pietarila-Graham, viz: VAPOR)

From: Alex Szalay, JHU

Daily Usage

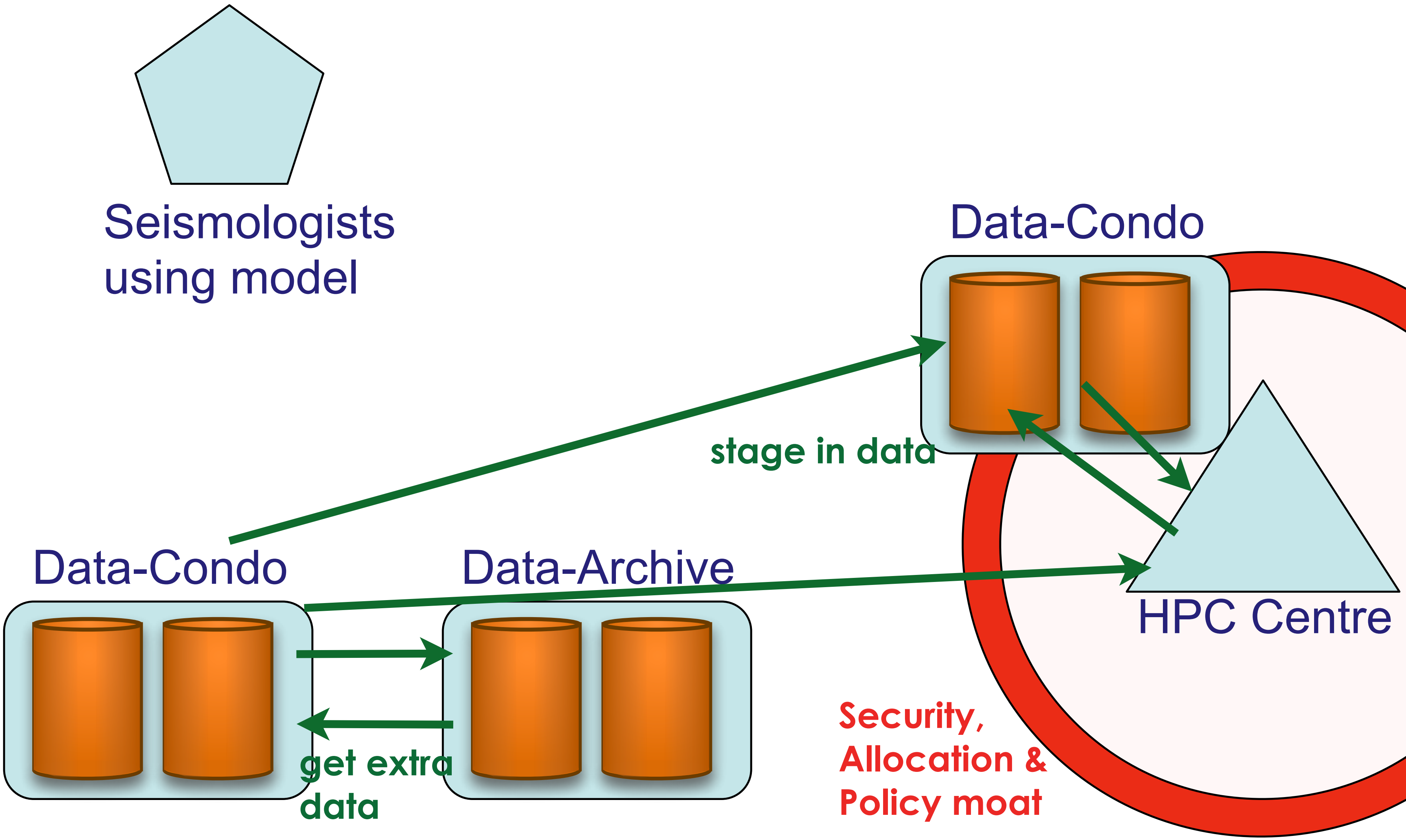


2011: exceeded 100B points, delivered publicly

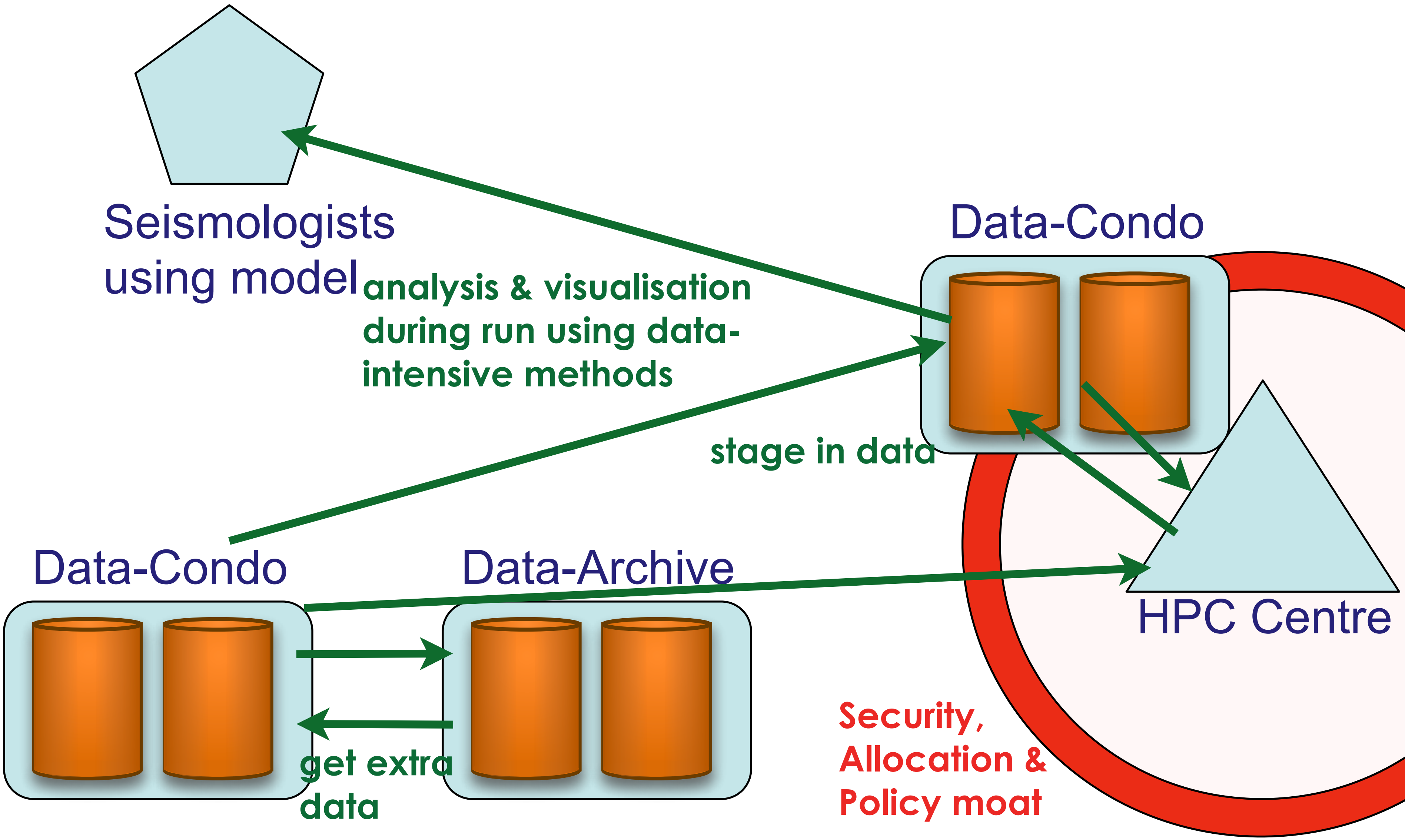
From: Alex Szalay, JHU

Data-Intensive and HPC in close harmony

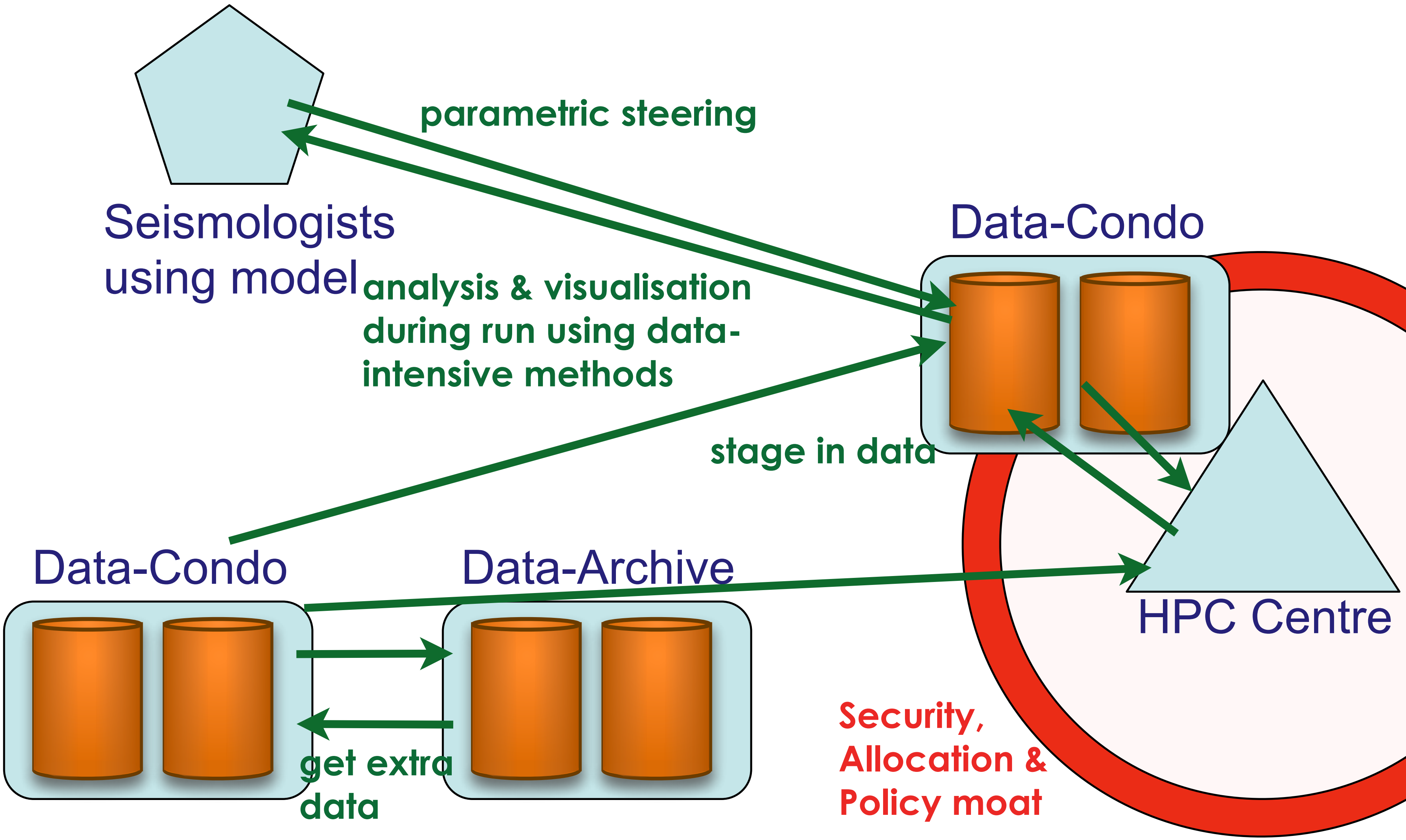
Closely coupled model: steer the model



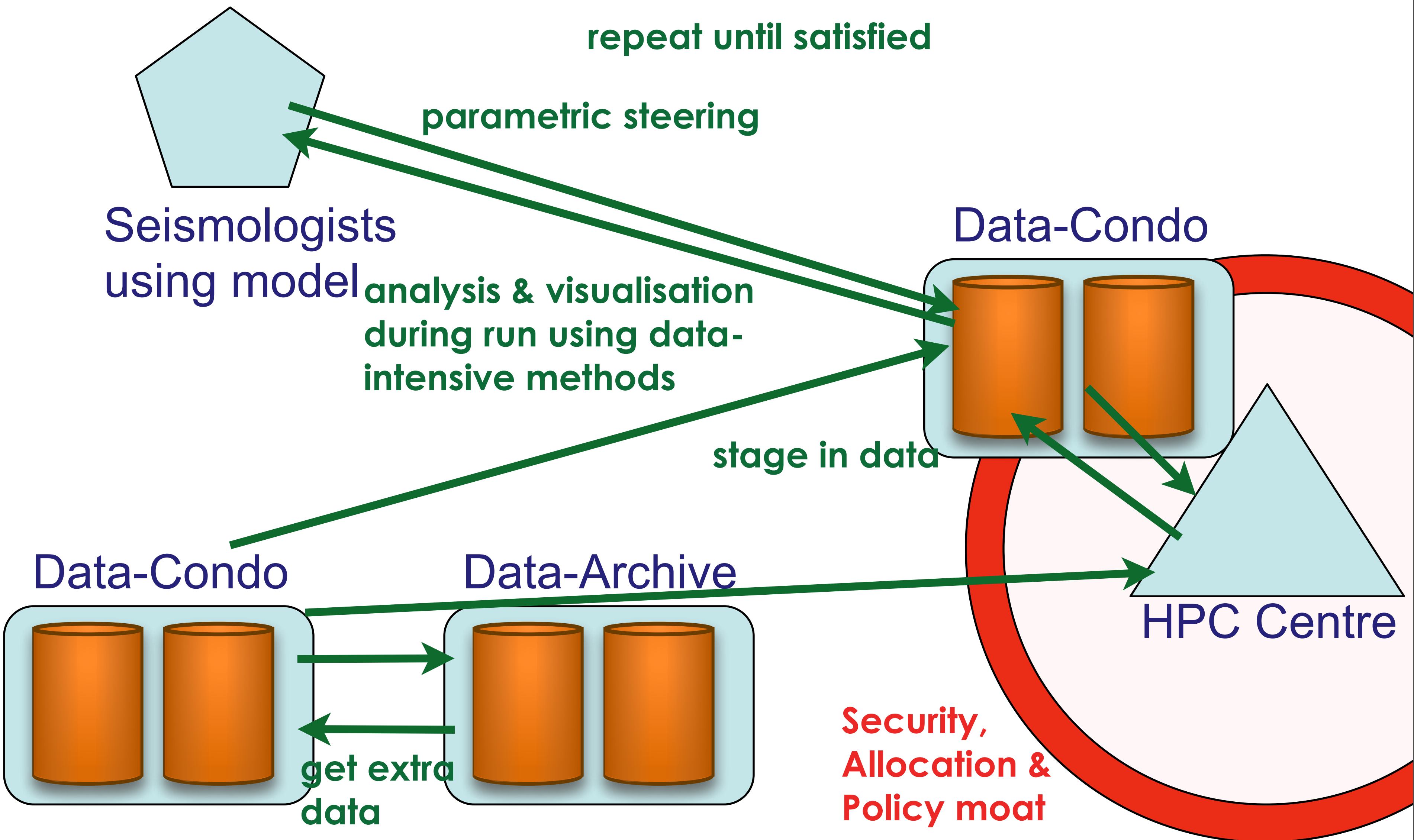
Closely coupled model: steer the model



Closely coupled model: steer the model

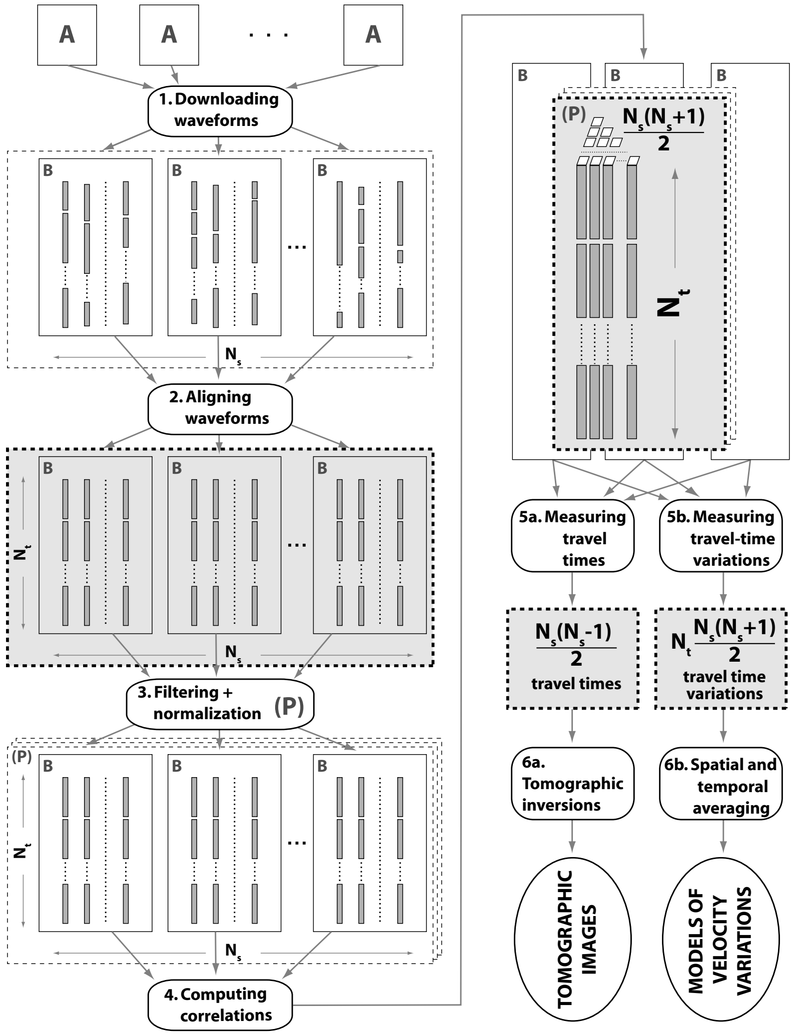


Closely coupled model: steer the model



Data-Intensive thinking about data-preparation & correlation

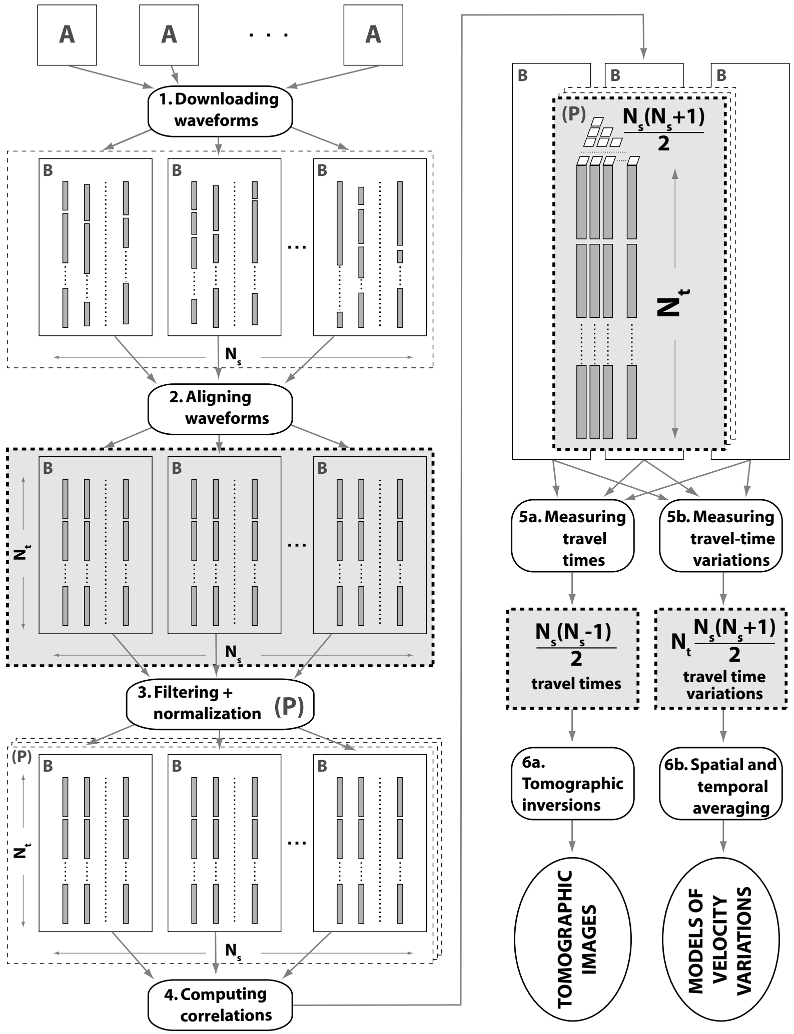
Data-Intensive View of Seismic noise correlation



- From Jean-Pierre Vilotte's talk
  - VERCE use case
- Show use of WF
- Show use of patterns
- Discuss optimisations
- Discuss mapping to existing technologies
- The limit of the lies is the truth
  - whistle-stop tour
  - adding reality and optimisation incrementally
- To understand
  - replay & discuss with DIR folk
  - read the book

THE DATA BONANZA: Improving Knowledge Discovery for Science, Engineering and Business, Atkinson *et al.*, Wiley 2012

Data-Intensive View of Seismic noise correlation

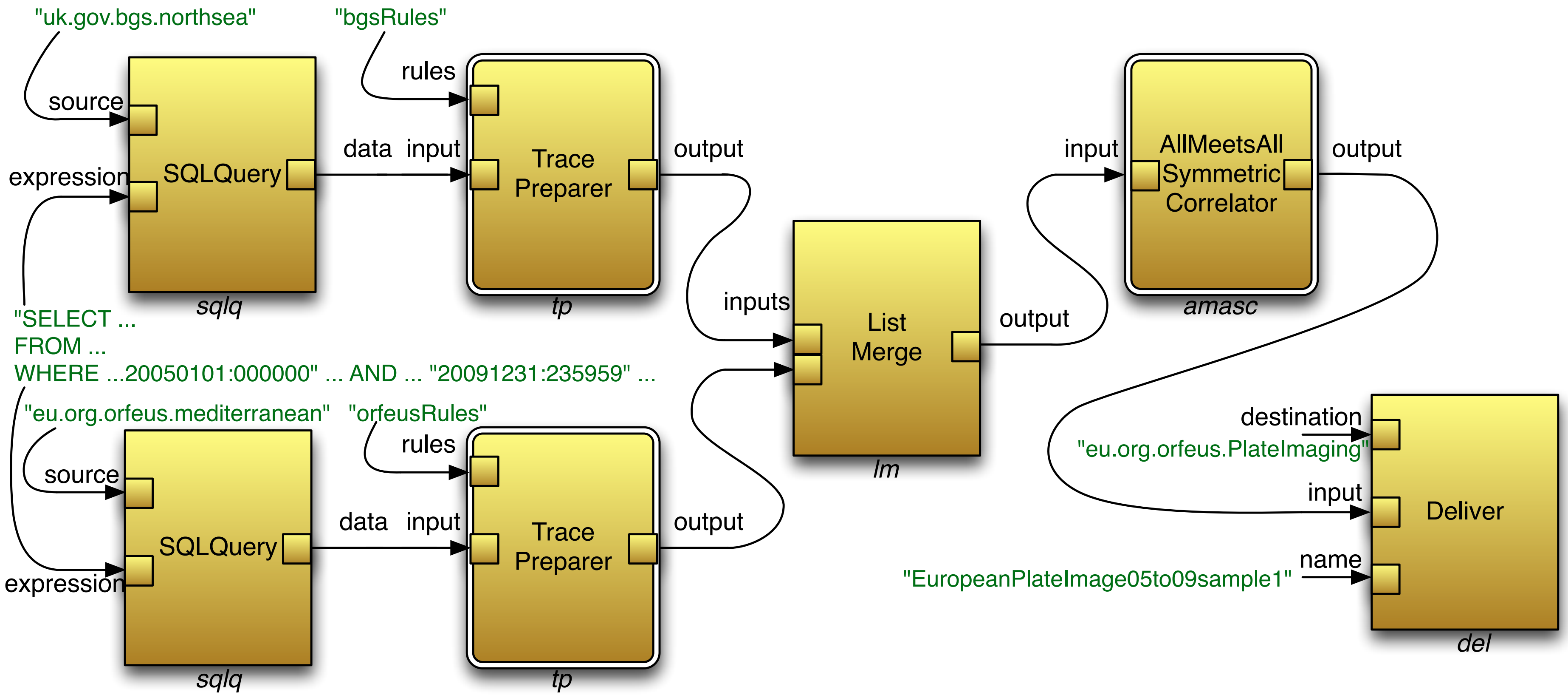


- From Jean-Pierre Vi
  - VERCE use case
- Show use of WF
- Show use of pattern
- Discuss optimisation
- Discuss mapping to existing technologies
- ~~The limit of the lies is the truth~~
  - whistle-stop tour
  - adding reality and optimisation incrementally
- To understand
  - replay & discuss with DIR folk
  - read the book

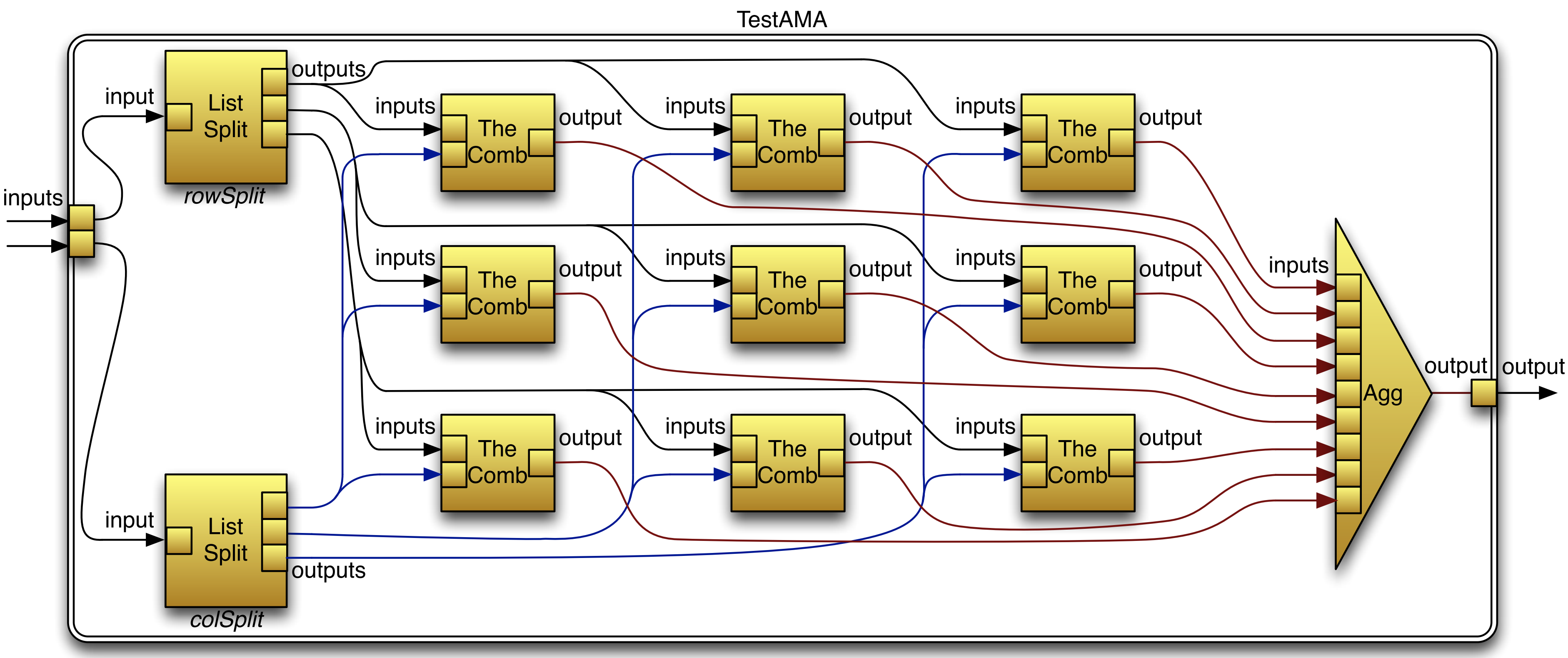
THE DATA BONANZA: Improving Knowledge Discovery for Science, Engineering and Business, Atkinson *et al.*, Wiley 2012

Unpredictable succession of insights & increments to definitions to improve and accelerate

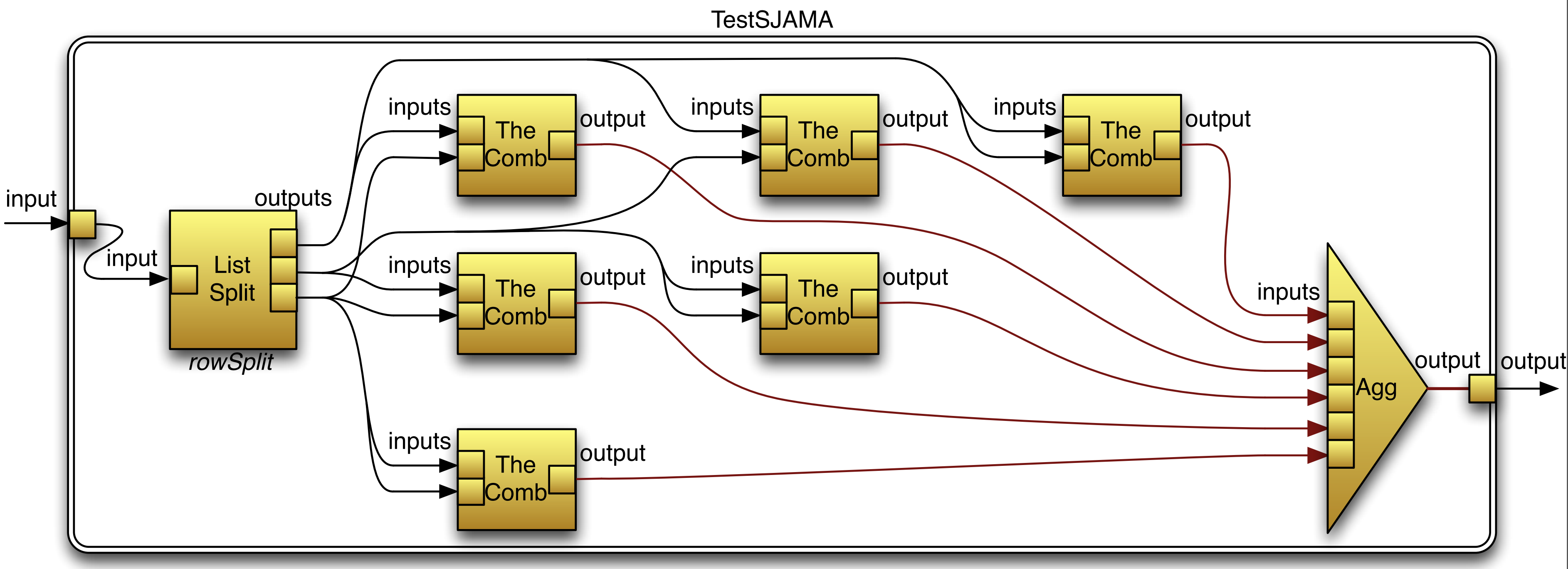
High-level WF generated via *science gateway*



Parallelising correlation

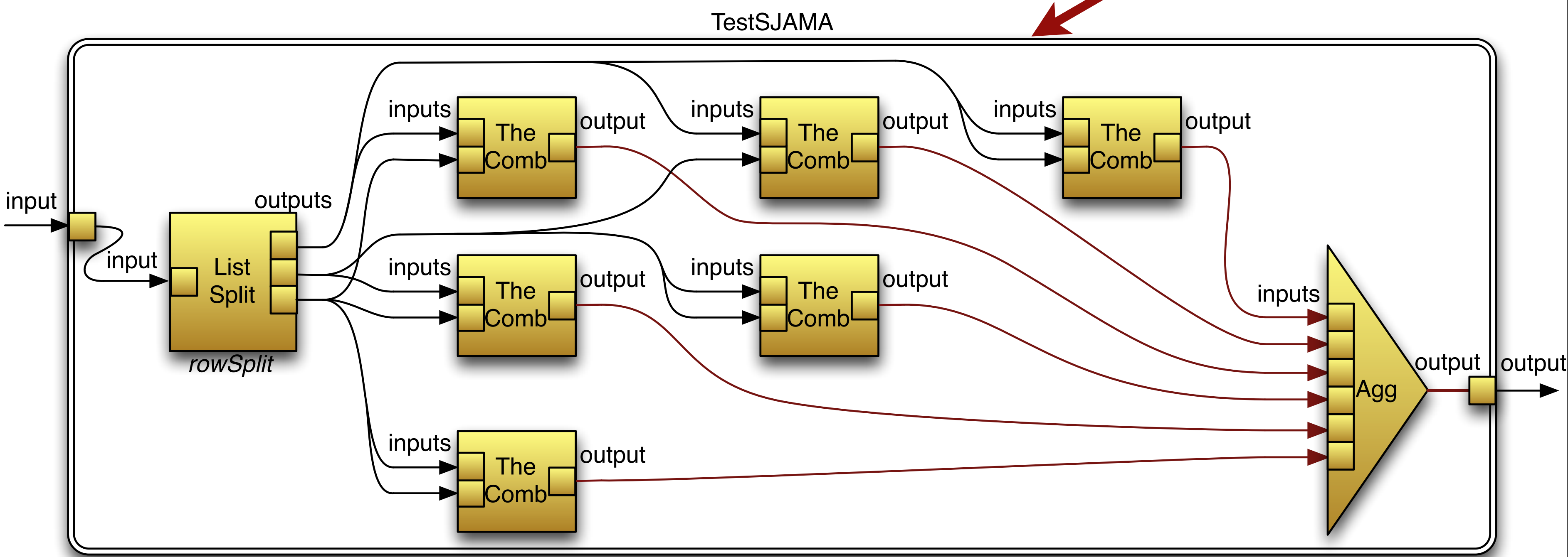


Parallelising symmetric correlation

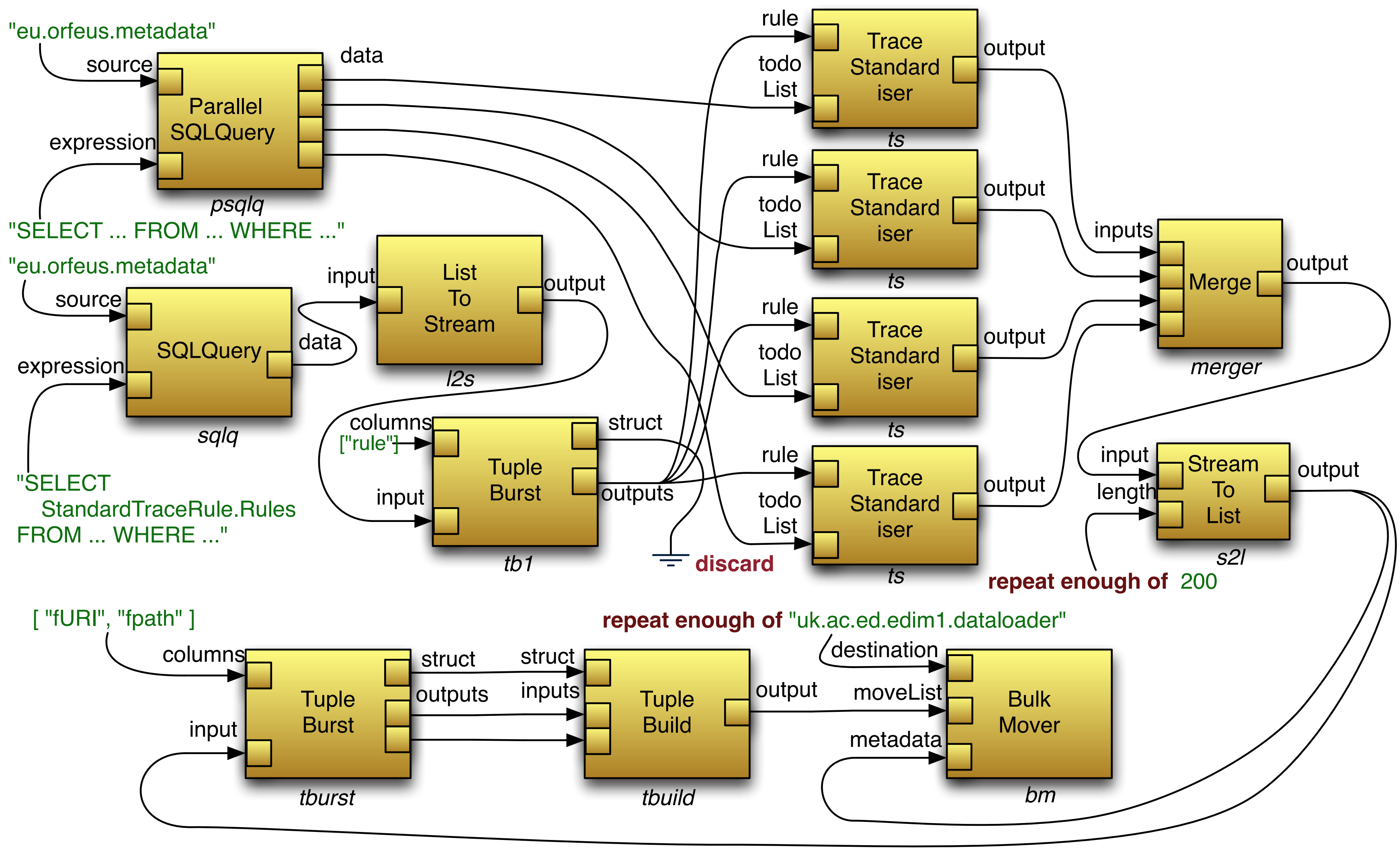


Parallelising symmetric correlation

actual number of nodes set by pattern generator to match data arrival rate

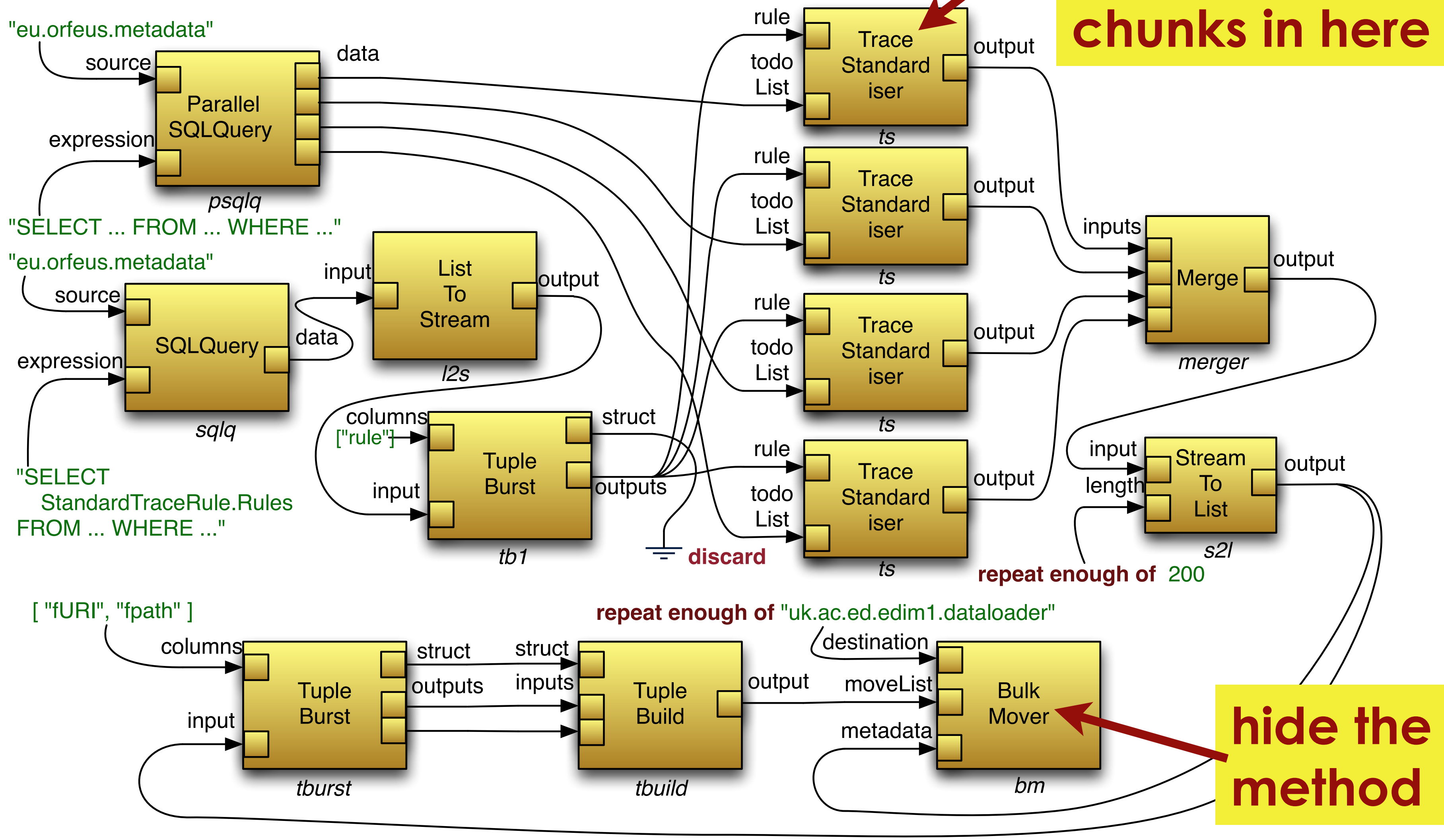


Parallel data preparation

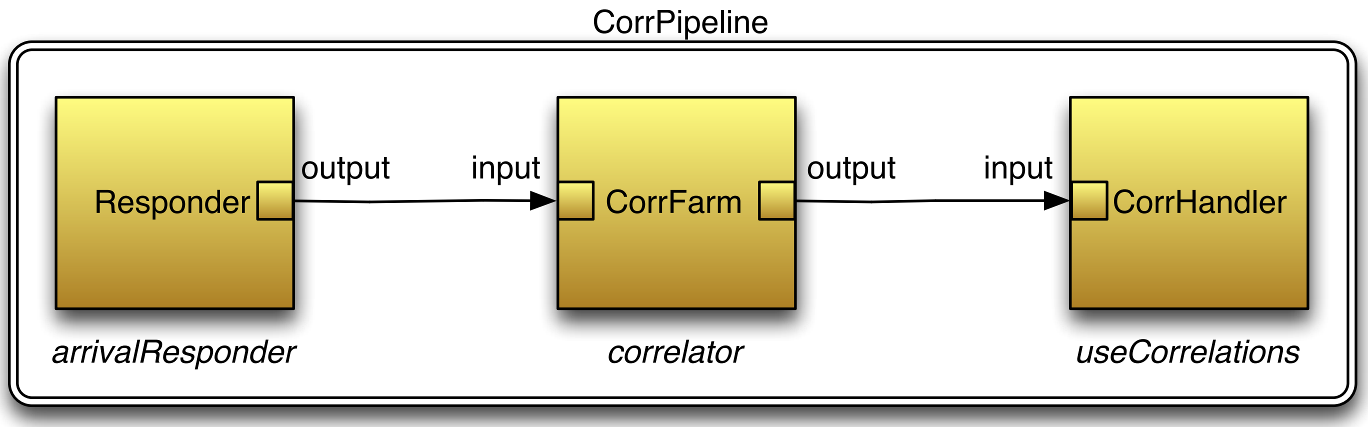


Parallel data preparation

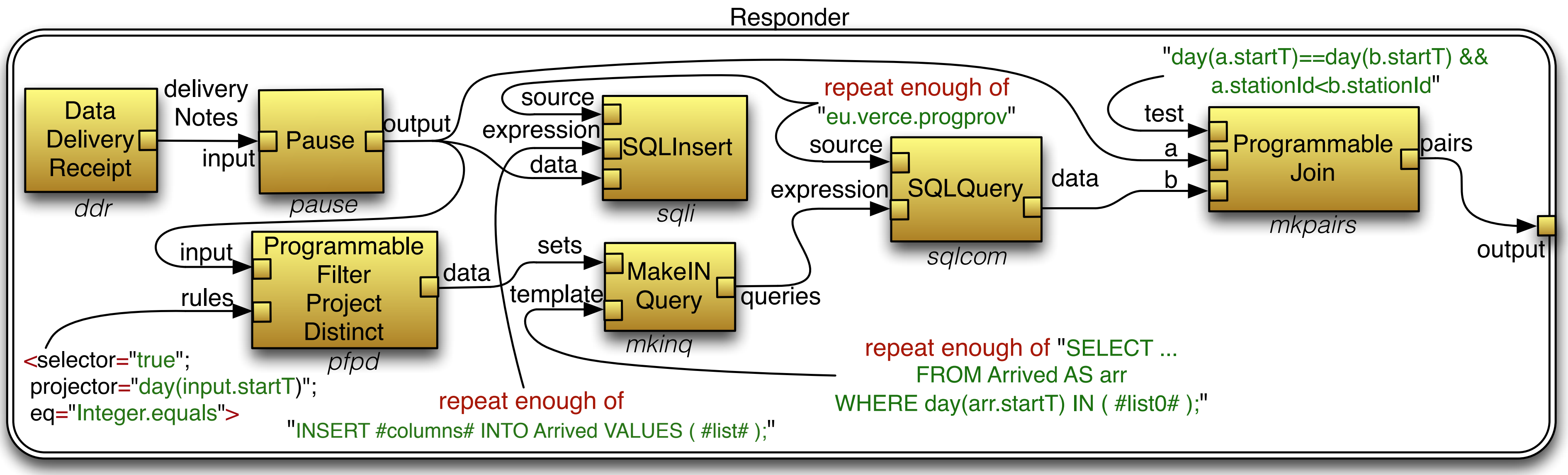
maximised pipelining with minimised chunks in here



High-level view of handling bulk-data deliveries

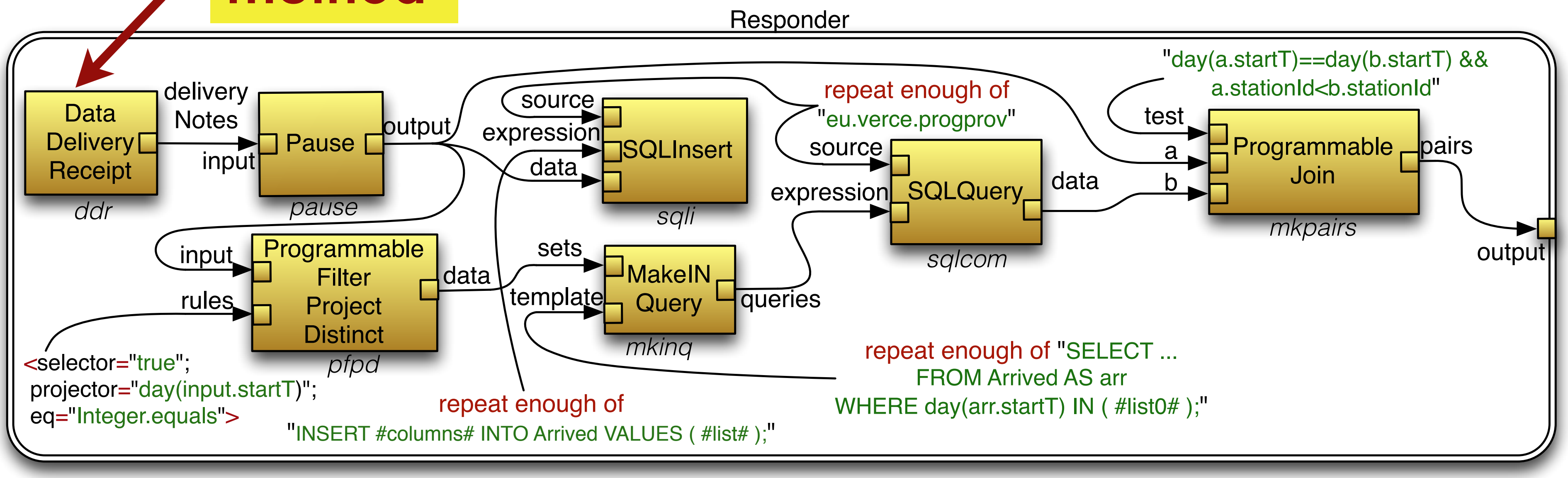


Keep records on arrival and locally distribute for correlation

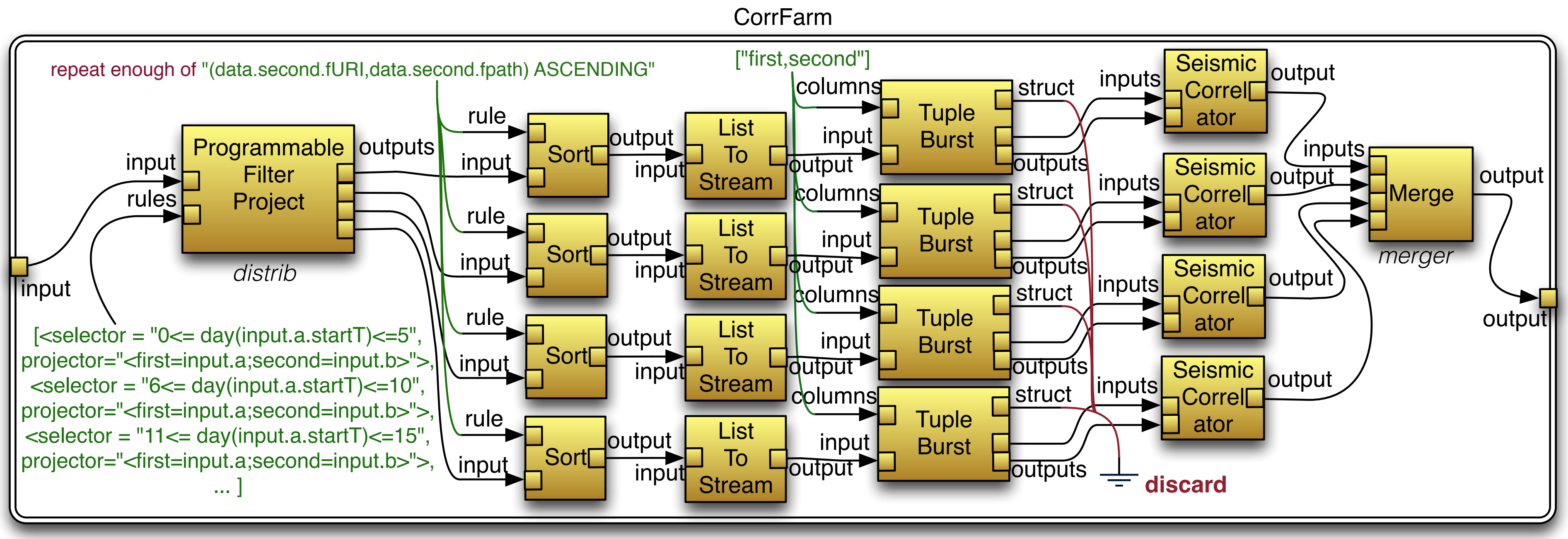


Keep records on arrival and locally distribute for correlation

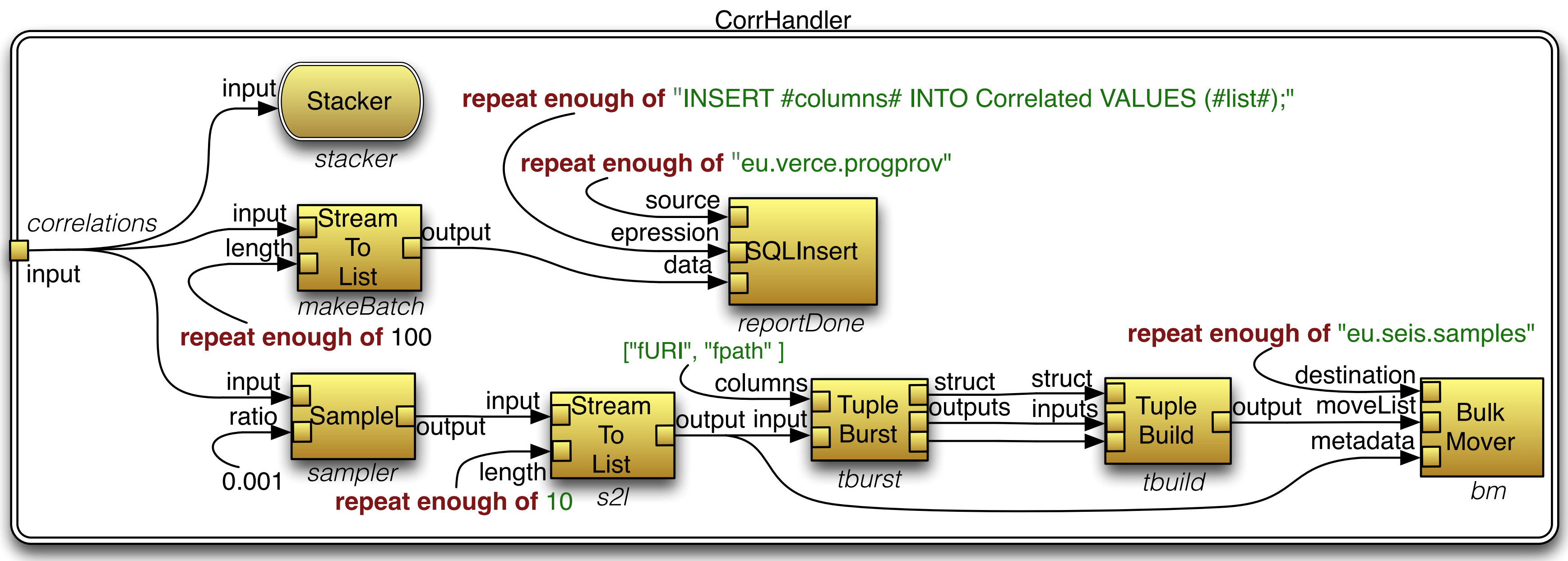
hide the method



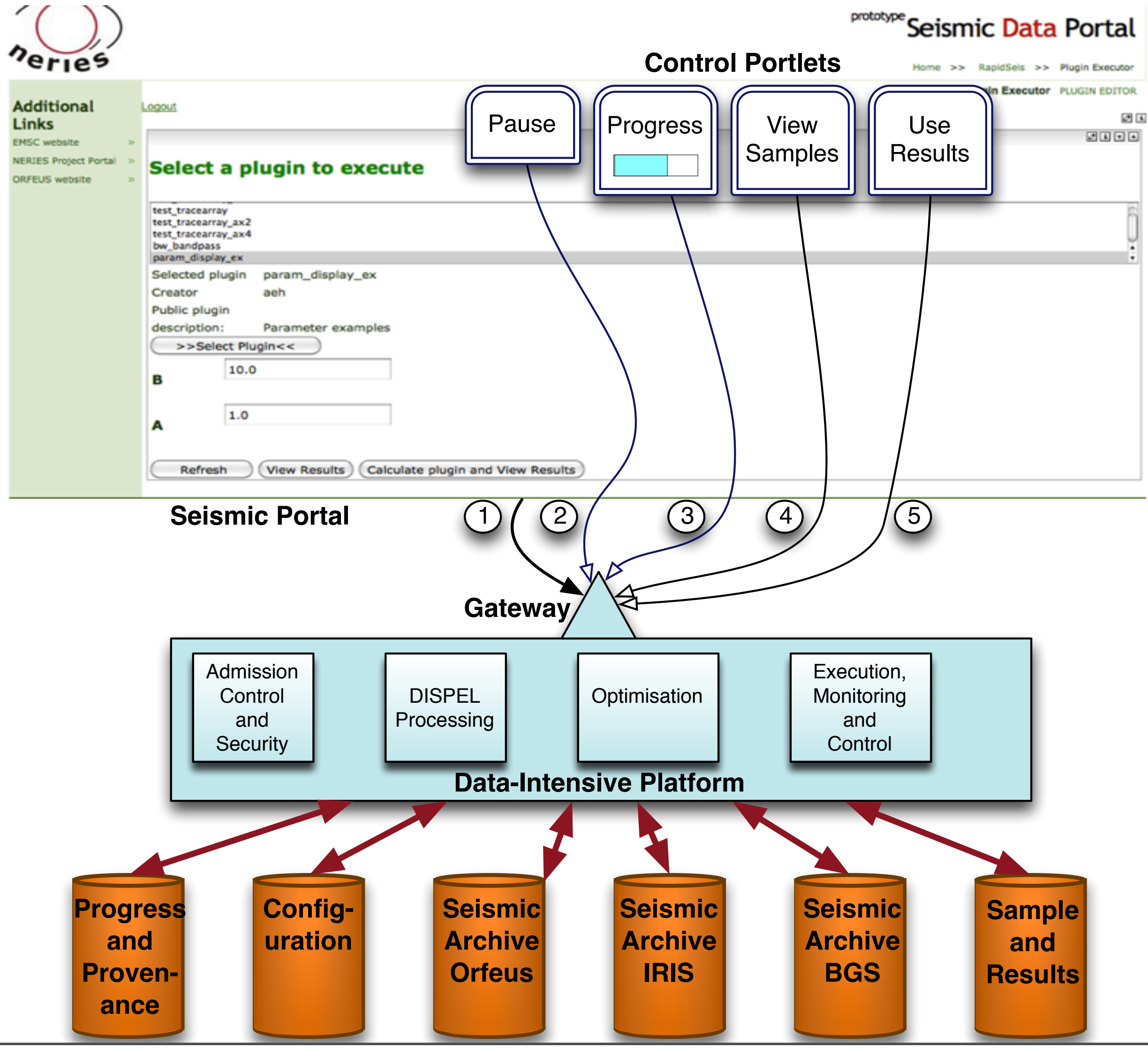
Parallel “hash-join” correlation farm



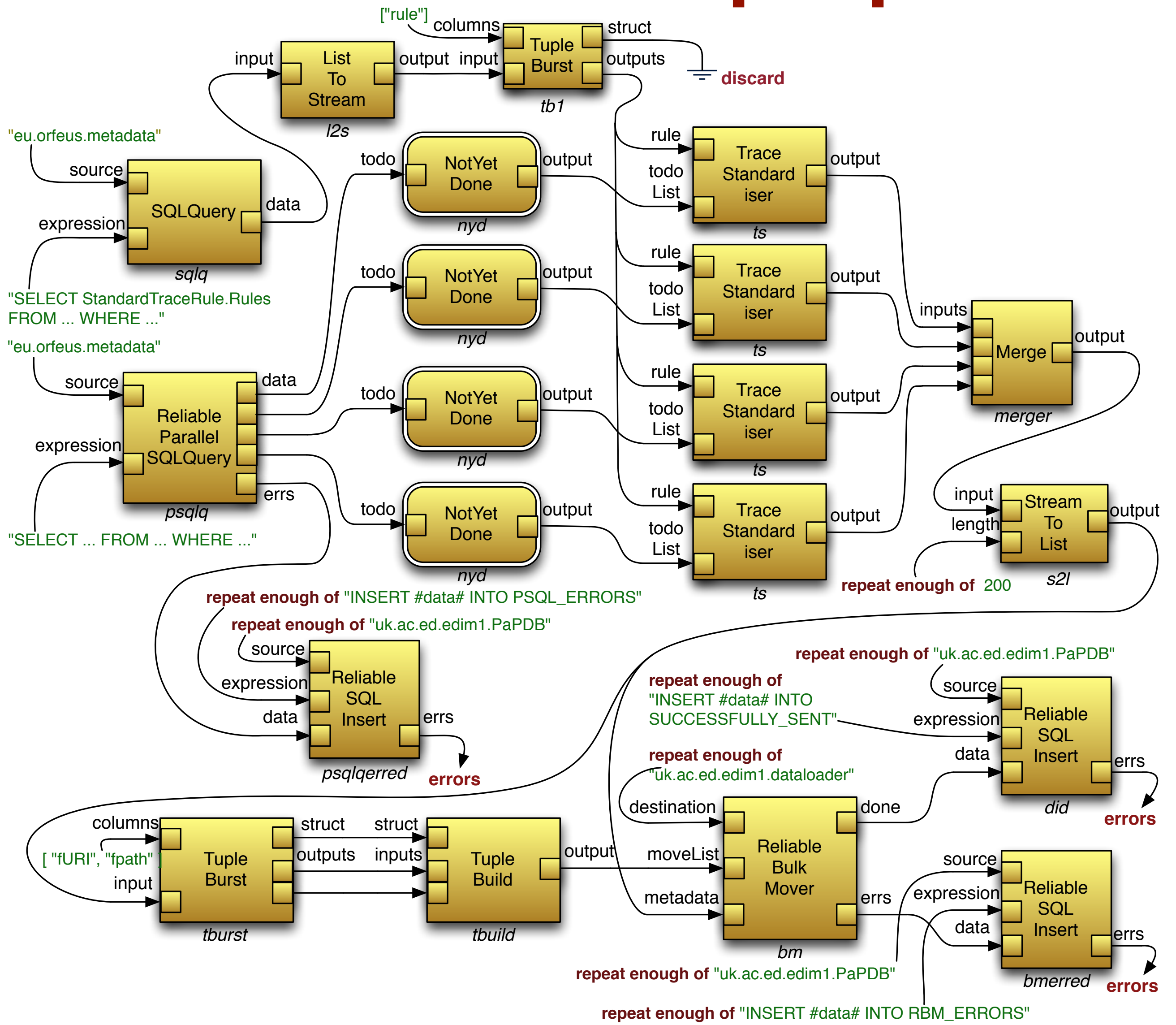
Keep track of progress & stack



The seismologist in control: providing the scientific cockpit



Add recovery of work after partial failure & incremental preparation



Summary and Conclusions

Summary

Summary

- VERCE is designing & prototyping e-Infrastructure for seismologists
  - with a continuity model based on wider use of the VERCE platform

Summary

- VERCE is designing & prototyping e-Infrastructure for seismologists
  - with a continuity model based on wider use of the VERCE platform
- Infrastructure is not *designed and then built*

Summary

- VERCE is designing & prototyping e-Infrastructure for seismologists
  - with a continuity model based on wider use of the VERCE platform
- Infrastructure is not *designed and then built*
- It evolves organically
  - driven by success, economics, investment and technological innovations
  - it is mostly shared with bespoke adaptations

Summary

- VERCE is designing & prototyping e-Infrastructure for seismologists
  - with a continuity model based on wider use of the VERCE platform
- Infrastructure is not *designed and then built*
- It evolves organically
  - driven by success, economics, investment and technological innovations
  - it is mostly shared with bespoke adaptations
- Today it's like Jules Verne's "*Around the world in 80 days*"
  - Growing number of good parts
  - You use your wits to cross the gaps

Summary

- VERCE is designing & prototyping e-Infrastructure for seismologists
  - with a continuity model based on wider use of the VERCE platform
- Infrastructure is not *designed and then built*
- It evolves organically
  - driven by success, economics, investment and technological innovations
  - it is mostly shared with bespoke adaptations
- Today it's like Jules Verne's "*Around the world in 80 days*"
  - Growing number of good parts
  - You use your wits to cross the gaps
- Your ceaseless challenge
  - Pick and ride the winers
  - Spot when to "move out of shipping and onto airlines"

Summary

- VERCE is designing & prototyping e-Infrastructure for seismologists
  - with a continuity model based on wider use of the VERCE platform
- Infrastructure is not *designed and then built*
- It evolves organically
  - driven by success, economics, investment and technological innovations
  - it is mostly shared with bespoke adaptations
- Today it's like Jules Verne's "*Around the world in 80 days*"
  - Growing number of good parts
  - You use your wits to cross the gaps
- Your ceaseless challenge
  - Pick and ride the winers
  - Spot when to "move out of shipping and onto airlines"

with help from IT friends, projects & international collaborations

Summary

there isn't a one-size fits all correct answer

- VERCE is designing & prototyping e-Infrastructure for seismologists
  - with a continuity model based on wider use of the VERCE platform
- Infrastructure is not *designed and then built*
- It evolves organically
  - driven by success, economics, investment and technological innovations
  - it is mostly shared with bespoke adaptations
- Today it's like Jules Verne's "*Around the world in 80 days*"
  - Growing number of good parts
  - You use your wits to cross the gaps
- Your ceaseless challenge
  - Pick and ride the winers
  - Spot when to "move out of shipping and onto airlines"

with help from IT friends, projects & international collaborations

Take home message

Take home message

- Infrastructure isn’t designed and then built

Take home message

- **Infrastructure isn't designed and then built**
- ***But it's components are!***
  - ***hundreds of person-years of design improving LINPAC (inherited in numpy)***
  - ***>50 accelerating linear file read & write***
  - ***>30 years polishing database algorithms***
  - ***>20 years improving data transport***
  - ***<10 years on large-scale map-reduce***

Take home message

- Infrastructure isn't designed and then built
- *But it's components are!*
  - *hundreds of person-years of design improving LINPAC (inherited in numpy)*
  - *>50 accelerating linear file read & write*
  - *>30 years polishing database algorithms*
  - *>20 years improving data transport*
  - *<10 years on large-scale map-reduce*

What will be your LINPAC for Data-Intensive Operations?