25 years of High Performance Computing: An Application Perspective

The 24th Conference on High-Speed Computing

Salishan April 19-22 2004

Geoffrey Fox
Computer Science, Informatics, Physics
Pervasive Technology Laboratories
Indiana University Bloomington IN 47401

gcf@indiana.edu
http://www.infomall.org http://www.grid2002.org

Personal Perspective I

- So I understood high performance computing in detail from 1980-1995
  - Hypercube, HPF, CRPC, Sundry Grand Challenges
  - I used HPT (Holes in Paper Tape) as an undergraduate
- I summarized applications for the "Source Book of Parallel Computing" in 2002 and had 3 earlier books
- I tried (and failed so far) to develop Java as a good high performance computing language from 1996-2003
- My last grant in this area developed HPJava (http://www.hpjava.org) but it was small and ended in 2003
- I have watched DoD scientists develop parallel code in their HPC Modernization Program
- I have worked closely with NASA on Earthquake Simulations from 1997-now

Personal Perspective II

- I have studied broadly requirements and best practice **biology** and **complex systems** (e.g. critical infrastructure and network) simulations
- Nearly all my research is nowadays in **Grid** (distributed) computing
- I have struggled to develop **computational science as an academic discipline**
- I taught **classes** in parallel computing and computational science to a dwindling audience with last one in 1998
  - DoD requests a rerun next fall via Access Grid
- I read **High End Computing Revitalization** reports/discussions and remembered **Petaflop** meetings of a decade ago

Some Impressions I

- Computational Science is highly successful with simulations in 1980 being "toy 2D explorations" and today we have full 3D multidisciplinary simulations with magnificent visualization
  - 128 node hypercube in 1983-5 had about 3 megaflop performance but it did run at 80% efficiency
  - Today DoE ASC machines and Earth Simulator can realize teraflop performance with 1-10% of peak speed
  - The whole talk can be devoted to descriptions of these simulations and their visualizations
- Some industry has adopted HPC (oil, drug) but runs at modest capability and most of action is in capacity computing and embarrassingly parallel computations (finance, biotech)
  - Aerospace is in between

Some Impressions II

- There is a group of users (such as those at this meeting) with HPC knowledge at their fingertips who can use current hardware with great effectiveness and maximum realistic efficiency
- I suspect in most fields, the knowledge of "average" users is at best an ability to use MPI crudely and their use of machines will be good only if they are wise enough to use good libraries like PetSc
  - Users seemed more sophisticated in 1980-95
  - "strategy for HPC" is different for new users and new applications
- Computer Science students (at universities I have been at) have little interest in algorithms or software for parallel computing
- Increasing gulf between the Internet generation raised on Python and Java and the best tools (Fortran, C, C++) of HPC
  - Matlab and Mathematica represent another disparate approach
  - Java Grande was meant to address this

Some Impressions III

- Situation today in HPC is not drastically different from that expected around 1985
  - Simulations getting larger in size and sophistication
  - Move from regular to adaptive irregular data structures
  - Growing importance of multidisciplinary simulations
  - Perhaps Moore's law has continued and will continue for longer than expected
  - Computation reasonably respected as a science methodology
- I expected more performance increase from explicit parallelism and less from more sophisticated chips
  - i.e. I expected all machines (PCs) to be (very) parallel and software like Word to be parallel
  - I expected 10^5 to 10^6-way not 10^4 way high end supercomputers with nCUBE/Transputer/CM2 plus weitek style architectures

Some Impressions IV

- So parallel applications succeeded roughly as expected but the manner was a little different
- As expected, essentially all scientific simulations could be parallelized and the CS/Applied Math/Application community has developed remarkable algorithms
  - As noted many scientists unaware of them today and some techniques like adaptive meshes and multipole methods are not easy to understand and use
  - Field so successful that has almost put itself out of business
- The parallel software model MPI is roughly the same as “mail box communication” system described say in 1980 memo by myself and Eugene Brookes
  - Even in 1980 we thought it pretty bad

Some Impressions V

- I always thought of parallel computing as a **map from an application through a model to a computer**
- I am surprised that modern HPC computer architectures do **not clearly** reflect **physical structure** of most applications
  - After all Parallel Computing Works because Mother Nature and Society (which we are simulating) are parallel
- **GRAPE** and earlier particle dynamics successfully match special characteristics (low memory, communication bandwidth) of $O(N^2)$ algorithms.

Of course **vectors** were introduced to reflect natural scientific data structures

Note Irregular problems still have geometrical structure even if no constant stride long vectors

Parallel Computing Works 1994

Basic Question → $S_1 \to S_2 \ldots \to S_{N-2}$ (numerical formulation)

→ S_{N-1} (numerical formulation modified for particular hardware, e.g., parallelism) $= S_{num}$

(3.1)

I think mismatch between hardware and problem architecture reflects software (Languages)

Low level software ⟶ S_N (Actual Computer) $= S_{comp}$

Some Impressions VI

- Two key features of today's applications
  - Is the simulation built on fundamental equations or phenomenological (coarse grained) degrees of freedom
  - Is the application deluged with interesting data
- Most of HPCC activity 1990-2000 dealt with applications like QCD, CFD, structures, astrophysics, quantum chemistry, neutron transport where reasonably accurate information available to describe basic degrees of freedom
- Classic model is to set up numerics of "well established equations" (e.g. Navier Stokes) and solve with known boundary values and initial conditions
- Many interesting applications today have unknown boundary conditions, initial conditions and equations
  - They have a lot of possibly streaming data instead
- For this purpose, the goal of Grid technology is to manage the experimental data

Data Deluged Science

- In the past, we worried about data in the form of parallel I/O or MPI-IO, but we didn't consider it as an enabler of new algorithms and new ways of computing
- Data assimilation was not central to HPCC
- ASC set up because didn't want test data!
- Now particle physics will get 100 petabytes from CERN
  - Nuclear physics (Jefferson Lab) in same situation
- Weather, climate, solid earth (EarthScope)
- Bioinformatics curated databases (Biocomplexity only 1000's of data points at present)
- Virtual Observatory and SkyServer in Astronomy
- Environmental Sensor nets

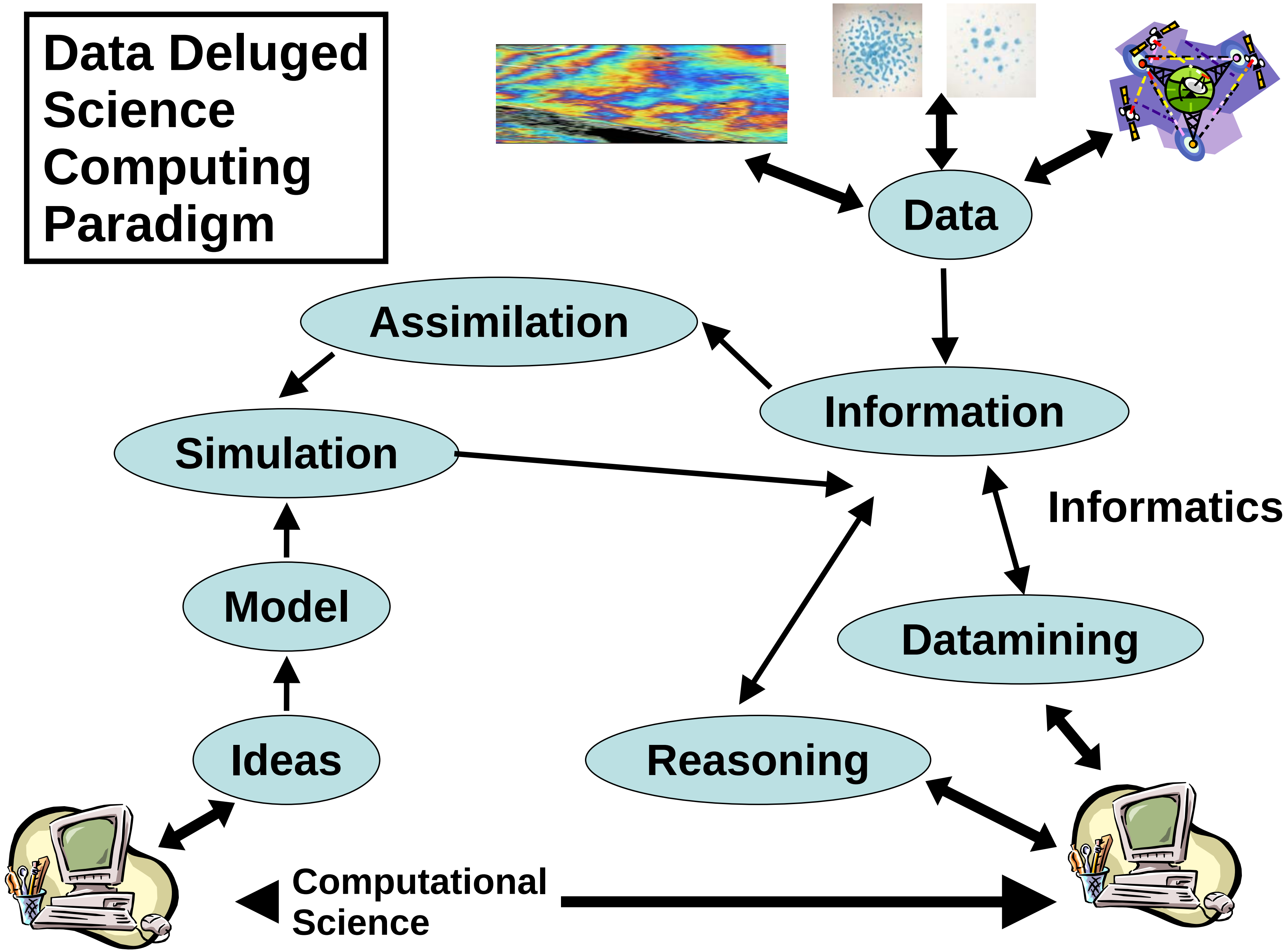
Computing Requirements for Weather

	2002 System	2010+ System	

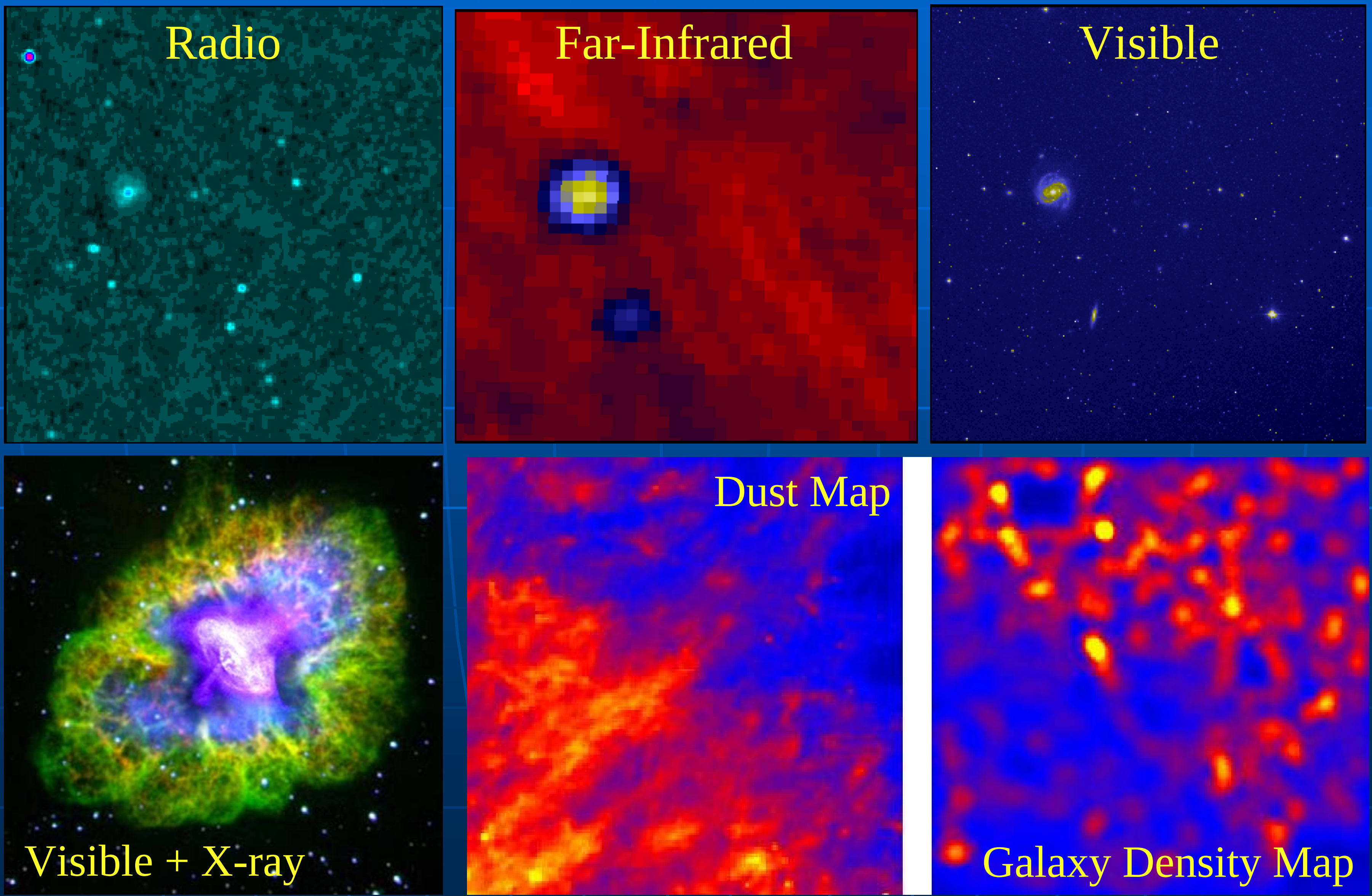
Resolution
• Horizontal
• Vertical levels
• Time step
• Observations
  ○ Ingested
  ○ Assimilated
	
	100 km
	55
	30 minutes
	
	10^7 / day
	10^5 / day
		
		10 km
		100
		6 minutes
		
		10^{11} / day
		10^8/ day	

System Components:	Atmosphere Land-surface Data assimilation	Atmosphere, Land-surface, Ocean, Sea-ice, Next-generation data assimilation Chemical constituents (100)	
Computing: • Capability (single image system) • Capacity (includes test, validation, reanalyzes, development)	 10 GFlops 100 GFlops	Must Have 20 TFlops (2000x) 400 TFlop (4000x)	Important 50 TFlops 1 PFlops
Data Volume: • Input (observations) • Output (gridded)	 400 MB / day 2 TB / day	 1 PB / day 10 PB / day	

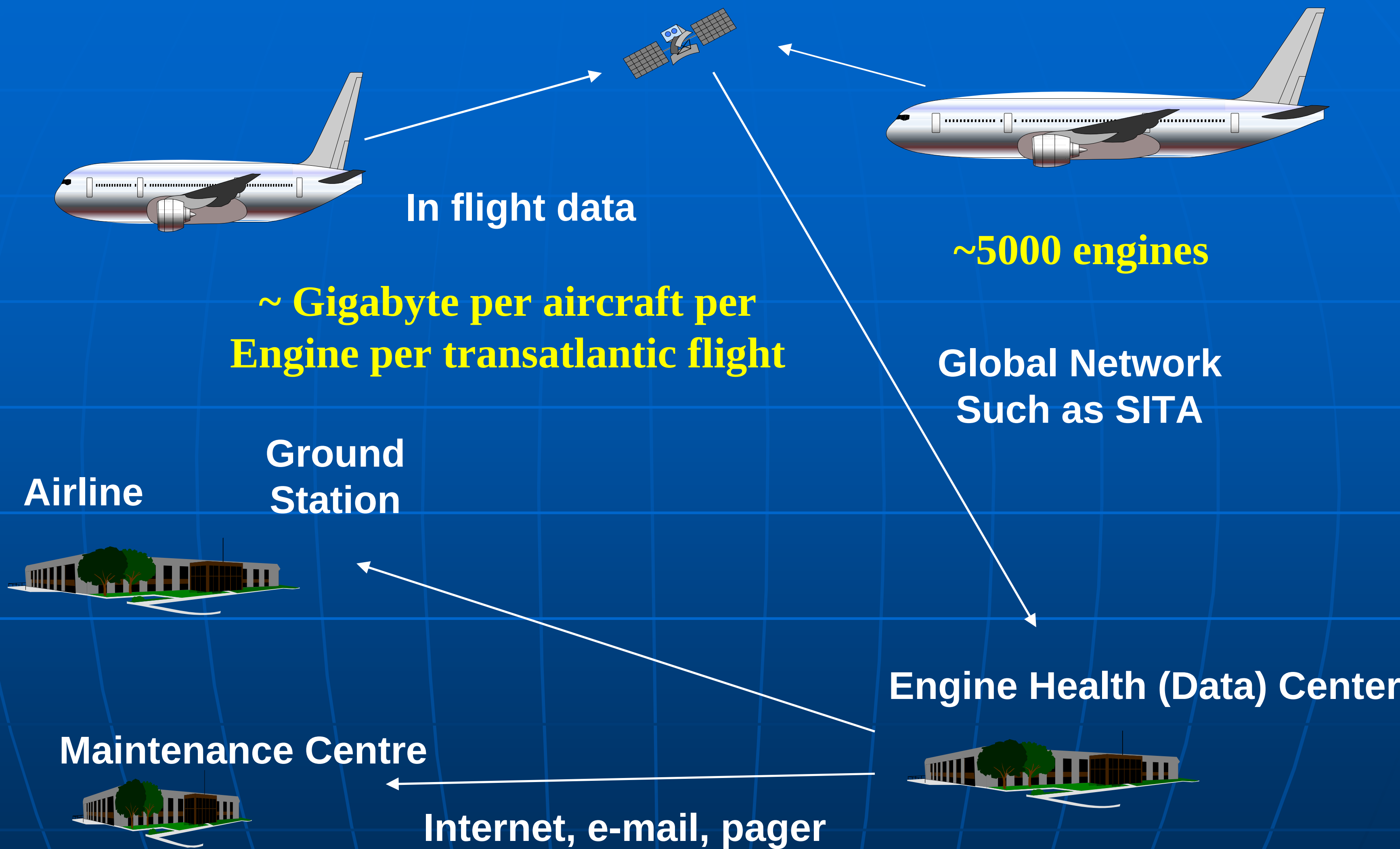
Networking/Storage
• Data movement
  ○ Internal
  ○ External
• Archival
	
	4 TB / day
	5 GB / day
	1 TB / day
		
		20 PB / day
		10 TB / day
		10 PB / day	


Data Deluged Science Computing Paradigm
Data
Assimilation
Information
Simulation
Informatics
Model
Datamining
Ideas
Reasoning
Computational Science

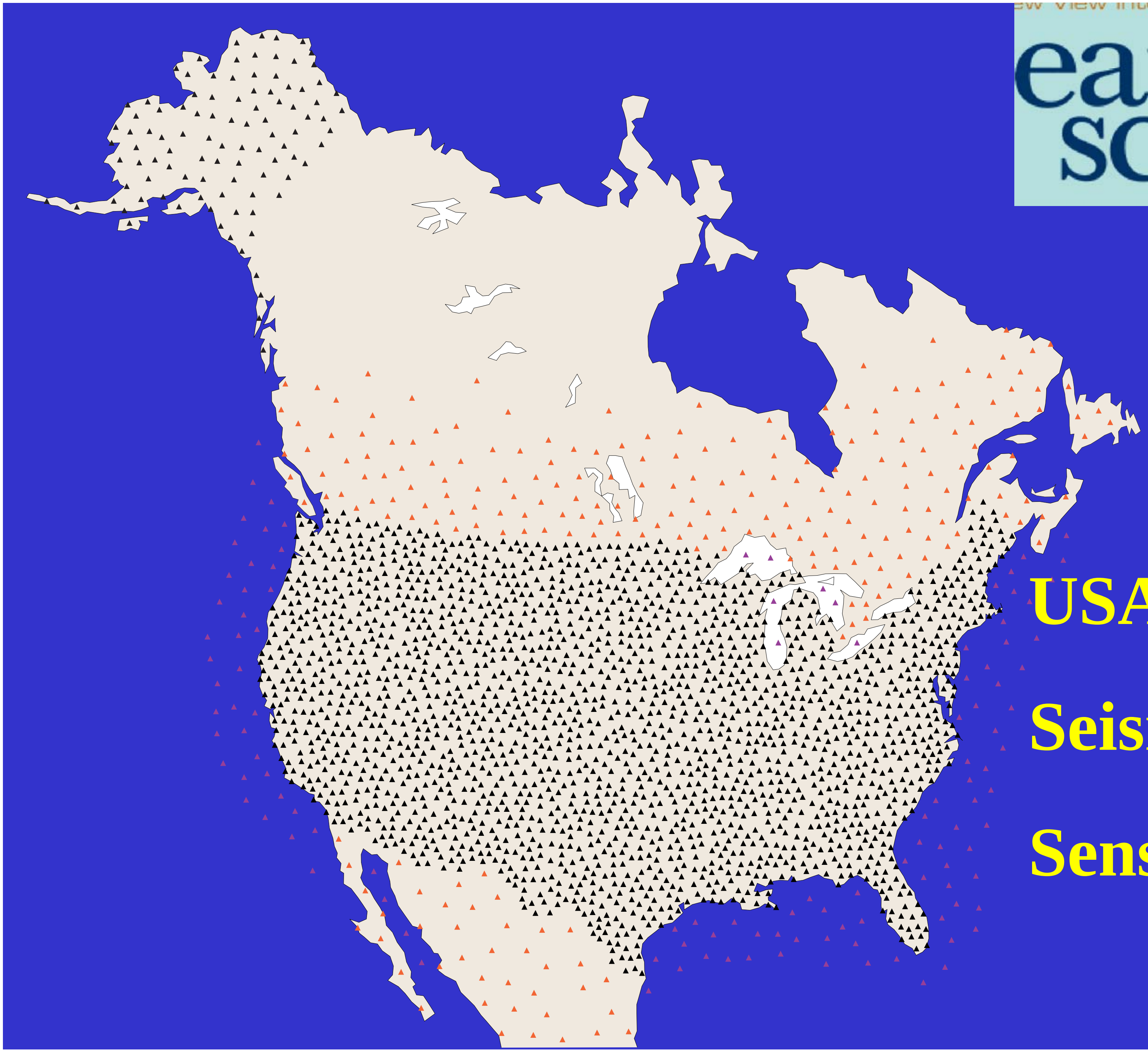
Virtual Observatory Astronomy Grid
Integrate Experiments



DAME Data Deluged Engineering



Rolls Royce and UK e-Science Program
Distributed Aircraft Maintenance Environment

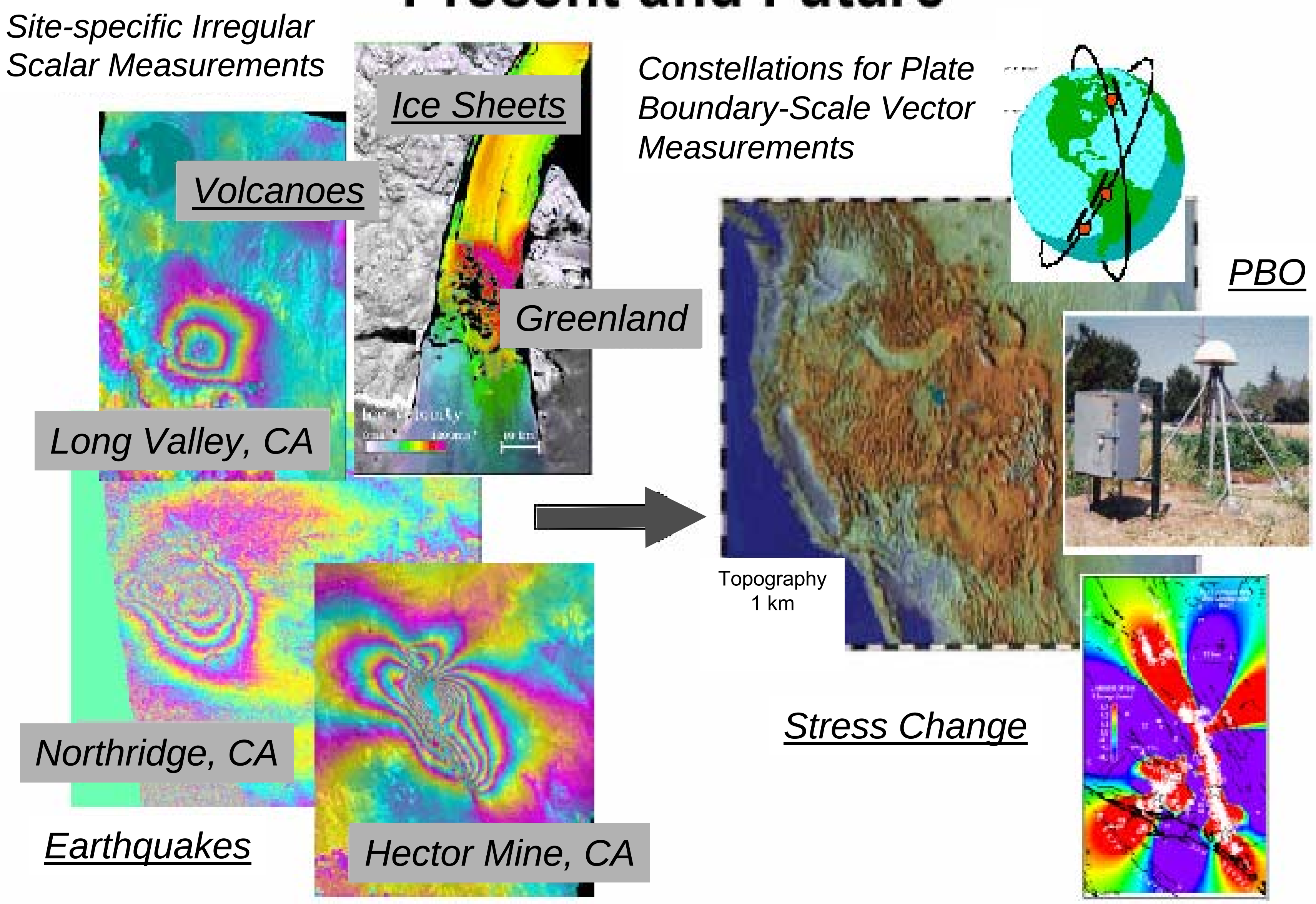


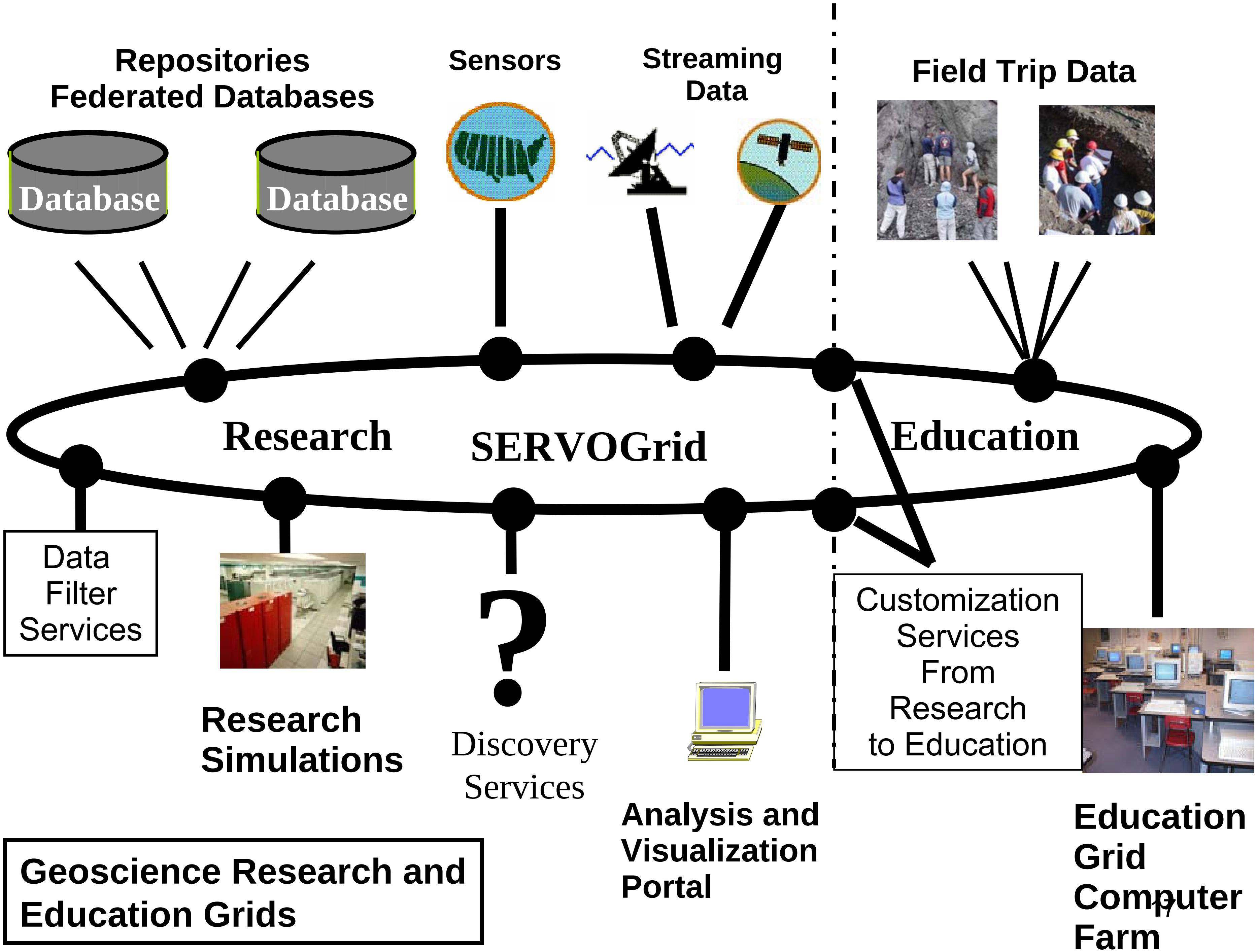
USArray

Seismic

Sensors

Interferometric SAR Observations: Present and Future




Repositories
Federated Databases
Database
Database
Sensors
Streaming
Data
Field Trip Data
Research
SERVOGrid
Education
Data
Filter
Services
Research
Simulations
?
Discovery
Services
Analysis and
Visualization
Portal
Customization
Services
From
Research
to Education
Education
Grid
Computer
Farm
Geoscience Research and
Education Grids

SERVOGrid Requirements

- **Seamless Access** to Data repositories and large scale computers
- **Integration** of **multiple data sources** including sensors, databases, file systems with analysis system
  - Including **filtered** OGSA-DAI (Grid database access)
- **Rich meta-data** generation and access with **SERVOGrid specific Schema** extending openGIS (Geography as a Web service) standards and using **Semantic Grid**
- **Portals** with component model for user interfaces and web control of all capabilities
- **Collaboration** to support world-wide work
- Basic Grid tools: **workflow** and **notification**
- **NOT metacomputing**

Non Traditional Applications: Biology

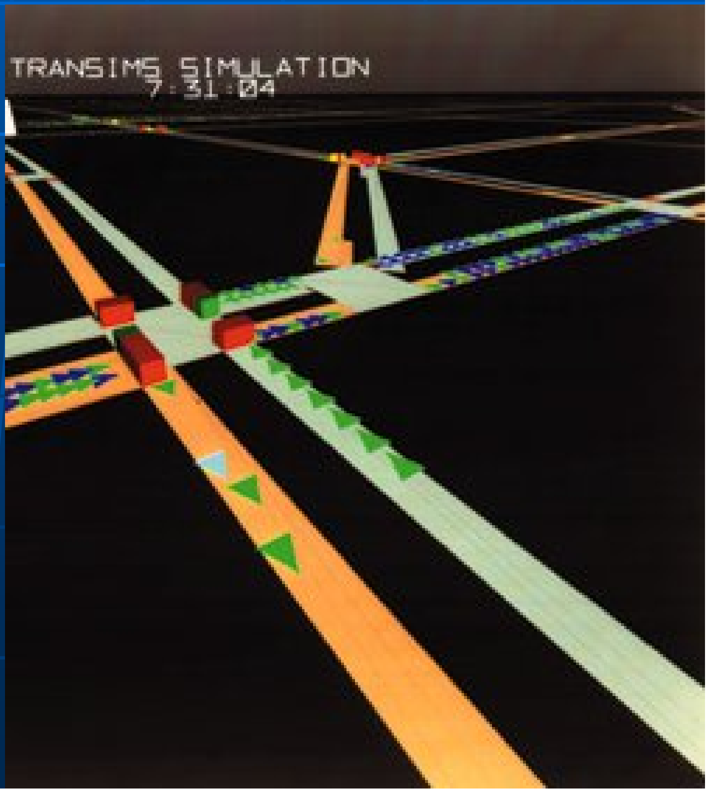
- At a fine scale we have molecular dynamics (protein folding) and at the coarsest scale CFD (e.g. blood flow) and structures (body mechanics)
- A lot of interest is in between these scales with
  - Genomics: largely pattern recognition or data mining
  - Subcellular structure: reaction kinetics, network structure
  - Cellular and above (organisms, biofilms) where cell structure matters: Cellular Potts Model
  - Neural Networks
- Data mining can be considered as a special case of a simulation where model is "pattern to be looked for" and data set determines "dynamics" (where the pattern is)

Non Traditional Applications: Earthquakes

- **We know the dynamics at a coarse level (seismic wave propagation) and somewhat at a fine scale (granular physics for friction)**
- **Unknown details of constituents and sensitivity of phase transitions (earthquakes) to detail, make it hard to use classical simulation methods to forecast earthquakes**
- **Data deluge (Seismograms, dogs barking, SAR) again does not directly tell you needed friction laws etc. needed for classic simulations**
- **Approaches like "pattern informatics" combine data mining with simulation**
  - **One is looking for "dynamics" of "earthquake signals" to see if the "big one" preceded by a certain structure in small quakes or other phenomenology**

Non Traditional Applications: Critical Infrastructure Simulations

- These include electrical/gas/water grids and Internet, transportation, cell/wired phone dynamics.
- One has some "classic SPICE style" network simulations in area like power grid (although load and infrastructure data incomplete)
  - 6000 to 17000 generators
  - 50000 to 140000 transmission lines
  - 40000 to 100000 substations
- Substantial DoE involvement through DHS

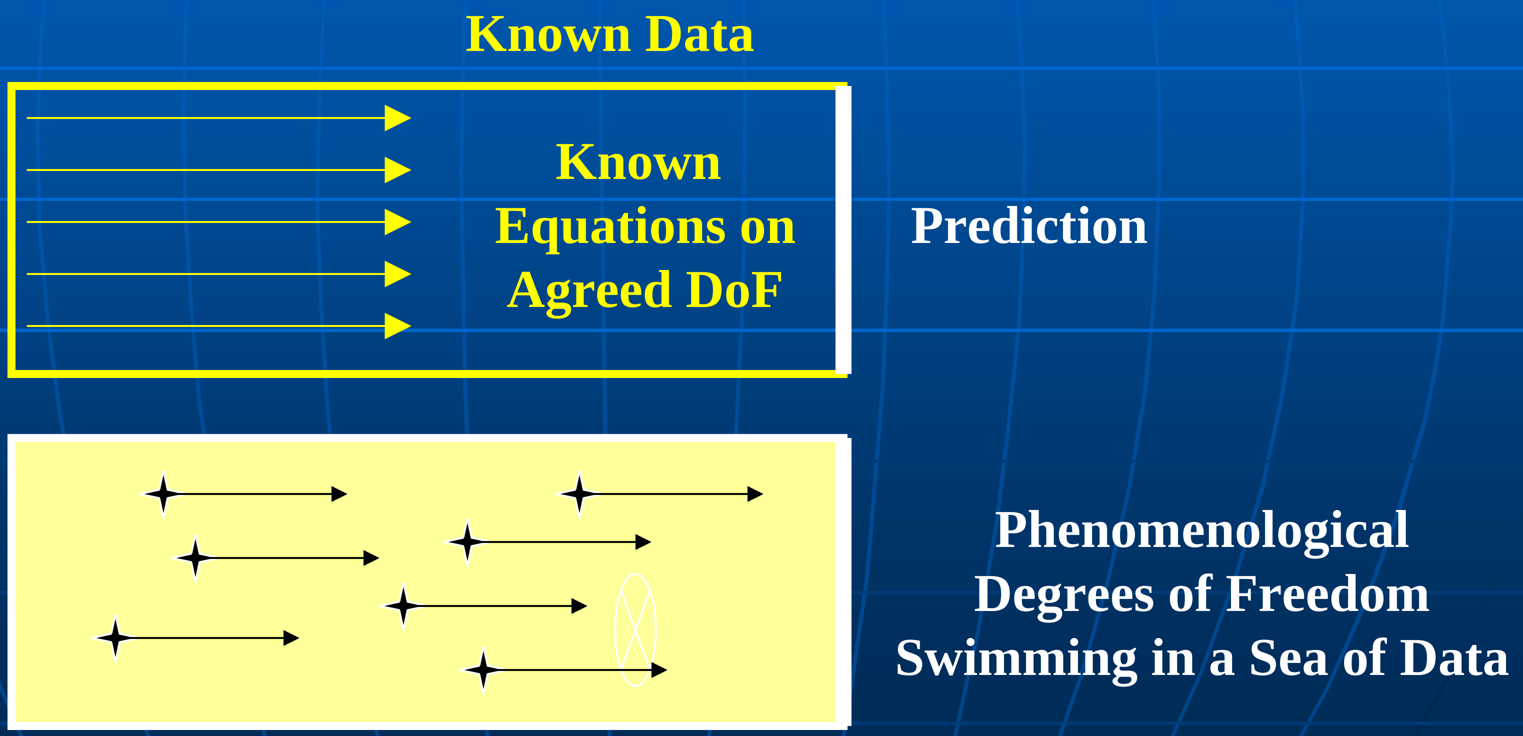


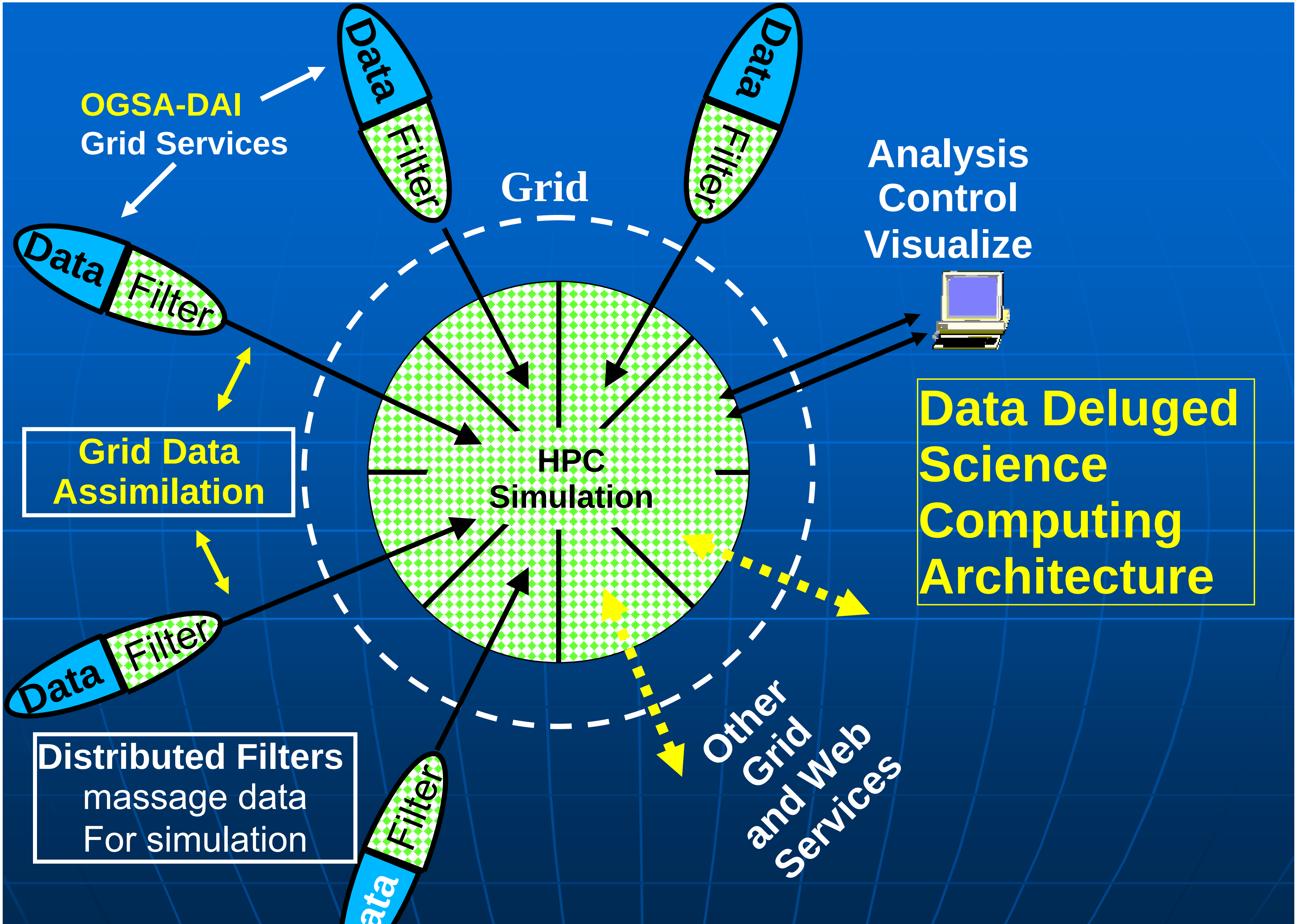
Non Traditional Applications: Critical Infrastructure Simulations

- **Activity data** for people/institutions essential for detailed dynamics but again these are not "classic" data but need to be "fitted" in **data assimilation** style in terms of some assumed lower level model.
  - They tell you goals of people but not their low level movement
- **Disease** and **Internet virus** spread and **social network** simulations can be built on dynamics coming from infrastructure simulations
  - Many results like "**small world**" internet connection structure are qualitative and unclear if they can be extended to detailed simulations
  - A lot of interest in (**regulatory**) **networks** in Biology

(Non) Traditional Structure

- **1) Traditional:** Known equations plus boundary values
- **2) Data assimilation:** somewhat uncertain initial conditions and approximations corrected by data assimilation
- **3) Data deluged Science:** Phenomenological degrees of freedom swimming in a sea of data





OGSA-DAI
Grid Services
Data
Filter
Data
Filter
Grid
Analysis
Control
Visualize
Data
Filter
HPC
Simulation
Grid Data
Assimilation
Data Deluged
Science
Computing
Architecture
Data
Filter
Other
Grid
and Web
Services
Distributed Filters
massage data
For simulation
Data
Filter

Data Assimilation

- **Data assimilation implies one is solving some optimization problem which might have Kalman Filter like structure**

$$\min_{Theoretical\ Unknowns} \sum_{i=1}^{N_{obs}} \left[Data_i(position, time) - Simulated\_Value \right]^2 \Big/ Error_i^2$$

- **Due to data deluge, one will become more and more dominated by the data (N_{obs} much larger than number of simulation points).**
- **Natural approach is to form for each local (position, time) patch the "important" data combinations so that optimization doesn't waste time on large error or insensitive data.**
- **Data reduction done in natural distributed fashion NOT on HPC machine as distributed computing most cost effective if calculations essentially independent**
  - **Filter functions must be transmitted from HPC machine**

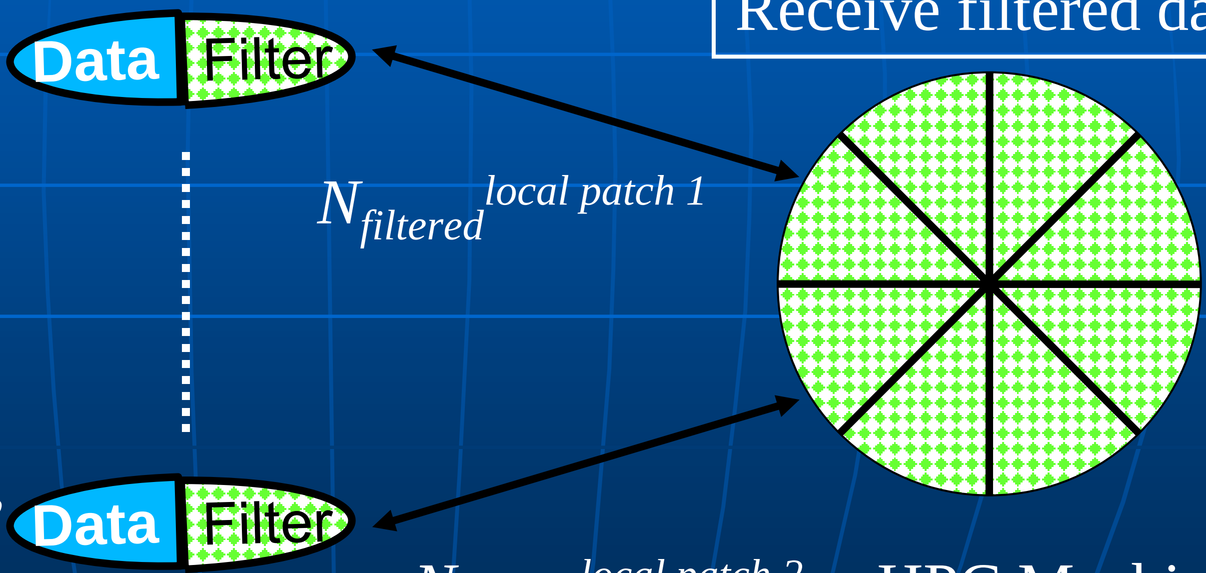
Distributed Filtering

$N_{obs}^{local\ patch} >> N_{filtered}^{local\ patch} \approx Number\_of\_Unknowns^{local\ patch}$

In simplest approach, filtered data gotten by linear transformations on original data based on Singular Value Decomposition of Least squares matrix

Send needed Filter
Receive filtered data

$N_{obs}^{local\ patch\ 1}$



$N_{filtered}^{local\ patch\ 1}$

Geographically
Distributed
Sensor patches

$N_{obs}^{local\ patch\ 2}$

$N_{filtered}^{local\ patch\ 2}$

HPC Machine

Factorize Matrix
to product of
local patches

Distributed
Machine

Some Questions for Non Traditional Applications

- No systematic study of how best to represent data deluged sciences without known equations
- Obviously data assimilation very relevant
- Role of Cellular Automata (CA) and refinements like the New Kind of Science by Wolfram
  - Can CA or Potts model parameterize any system?
- Relationship to back propagation and other neural network representations
- Relationship to "just" interpolating data and then extrapolating a little
- Role of Uncertainty Analysis – everything (equations, model, data) is uncertain!
- Relationship of data mining and simulation
- A new trade-off: How to split funds between sensors and simulation engines

Some Impressions VII

- My impression is that the knowledge of how to use HPC machines effectively is not broadly distributed
  - Many current users less sophisticated than you were in 1981
- Most simulations are still performed on sequential machines with approaches that make it difficult to parallelize
  - Code has to be re-engineered to use MPI
- The parallel algorithms in new areas are not well understood even though they are probably similar to those already developed
  - Equivalent of multigrid (multiscale) not used – again mainly due to software engineering issues – it's too hard
- Trade-off between time stepped and event driven simulations not well studied for new generation of network (critical infrastructure) simulations.

Some Impressions VIII

- I worked on Java Grande partly for obvious possible advantages of Java over Fortran/C++ as a language but also so HPC could better leverage the technologies and intellectual capital of the Internet generation
- I still think HPC will benefit from
- A) Building environments similar to those in the Internet world
  - Why would somebody grow up using Internet goodies and then switch to Fortran and MPI for their "advanced" work
- B) Always asking when to use special HPC and when commodity software/architectures can be used
  - Python often misused IMHO and standards like HPF, MPI don't properly discuss hybrid HPC/Commodity systems and their relation
  - The rule of the Millisecond
- I still think new languages (or dialects) that bridge simulation and data, HPC and commodity world are useful

Interaction of Commodity and HPC Software and Services

- Using commodity hardware or software obviously
  - Saves money and
  - Broadens community that can be involved e.g. base parallel language on Java or C# to involve the Internet generation
- Technologies roughly divide by communication latency
  - I can get high bandwidth in all cases?
  - e.g. Web Services and SOAP can use GridFTP and parallel streams as well as slow HTTP protocols
- >1 millisecond latency: message based services
- 10-1000 microseconds: method based scripting
- 1-20 microseconds: MPI
- < 1 microsecond: inlining, optimizing compilers etc.?
- To maximize re-use and eventual productivity, use the approach with highest acceptable latency
  - Only 10% of code is the HPC part?