Scaling to the Petascale: Context and Workshop Review

BROUGHT TO YOU BY
Bob Wilhelmson

bw@ncsa.uiuc.edu

WHO AM I?

- Graduate student in computer science at Illinois working with ILLIAC IV, one of the first parallel computers
- Atmospheric scientist using high performance computing to study severe weather since the late 1960's when three dimensional storm modeling became possible on small grids (25 x 25 x 20 spatial grid points)
- Today I am working with a number of other researchers to prepare for using Blue Waters to understand tornado formation, evolution, and demise using 10 m resolution using 10,000 x 10,000 x 2,000 grid points (100 x 100 x 20 km domain)
    - That is 10^8 times more grid points since the mid 1970's

WHO AM I?

- Professor, UIUC in Atmospheric Sciences
- Chief Scientist, NCSA
  - Co-PI on original unsolicited proposal in the mid 1980's to form NCSA
- Director, CyberApplications and Communities, NCSA
- Technical Advisory Committee member, Blue Waters
- Head of the storm modeling research group in the Department of Atmospheric

WHO AM I?

A VISUAL STORY

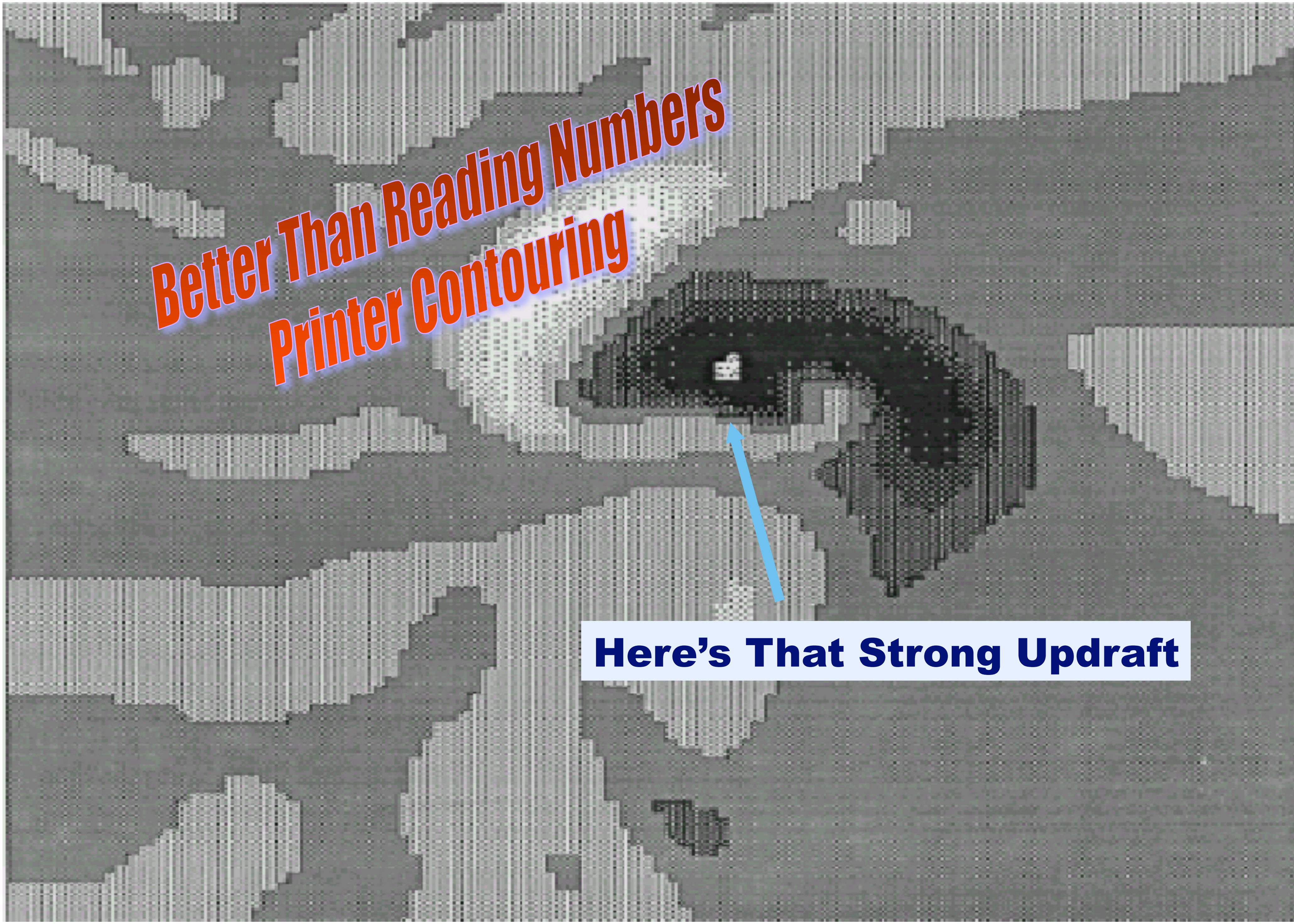
Thanks to all my research partners, students, post-docs

T= 6600. W AY FIELD AT SLAB 5

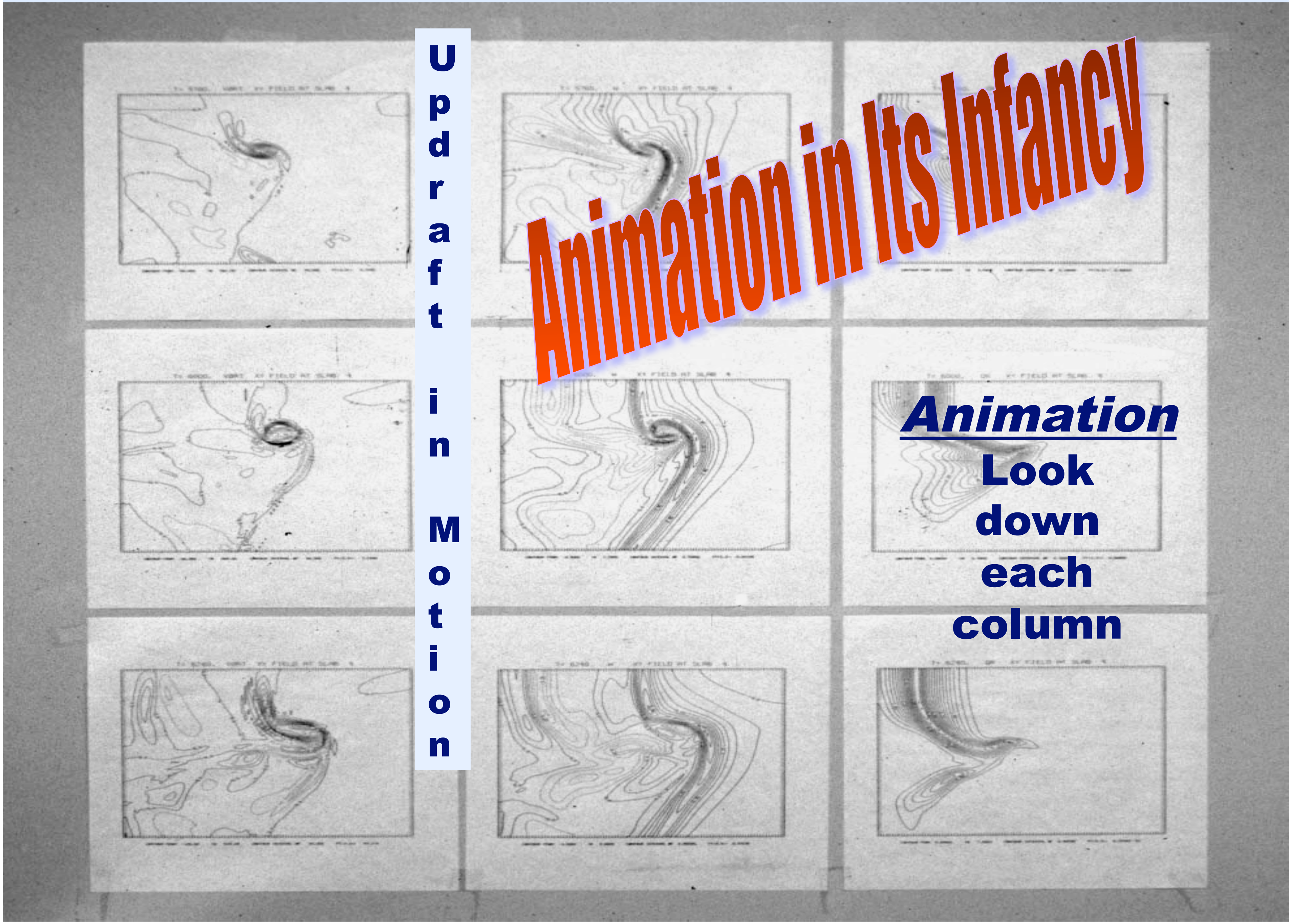
0.41	0.42	-0.19	-1.24	-0.74	0.03	0.20	0.07	-0.01	0.00	0.04	0.08	0.09	0.08	0.06
0.38	0.39	-0.25	-1.35	-0.80	0.00	0.17	0.07	0.01	0.04	0.11	0.18	0.21	0.18	0.15
0.38	0.37	-0.24	-1.39	-0.82	-0.01	0.19	0.12	0.10	0.14	0.29	0.42	0.45	0.39	0.34
0.40	0.39	-0.17	-1.34	-0.81	0.01	0.25	0.21	0.21	0.32	0.54	0.66	0.55	0.43	0.42
0.42	0.42	0.00	-1.52	-0.95	0.04	0.33	0.31	0.36	0.56	0.75	0.57	0.23	0.19	0.30
0.44	0.45	0.13	-1.35	-1.07	0.01	0.37	0.37	0.54	0.79	0.71	0.11	-0.14	0.05	0.24
0.47	0.49	0.24	-0.86	-1.08	0.01	0.38	0.44	0.69	0.83	0.37	-0.13	0.04	0.39	0.50
0.52	0.57	0.37	-0.56	-1.04	0.06	0.40	0.39	0.59	0.49	0.01	-0.04	0.42	0.81	0.98
0.61	0.67	0.56	-0.36	-1.01	-0.24	0.10	0.18	0.26	0.06	-0.13	0.10	0.04	1.18	1.11
0.72	0.81	0.81	-0.12	-1.70	-1.20	-0.58	-0.11	-0.10	[illegible]	[illegible]	[illegible]	[illegible]	1.24	0.04
0.86	0.96	1.03	0.34	-1.99	-2.12	-1.03	-0.39	0.17	[illegible]	[illegible]	[illegible]	[illegible]	1.06	0.36
0.99	1.10	1.12	0.87	2.08	4.16	6.30	7.83	10.39	[illegible]	[illegible]	[illegible]	[illegible]	1.56	1.22
1.08	1.15	1.16	0.84	7.46	13.67	14.84	16.31	19.01	[illegible]	[illegible]	[illegible]	[illegible]	1.69	-0.17
1.10	1.12	1.12	0.93	6.98	15.63	18.55	20.75	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	0.31	0.37
1.06	1.05	0.95	0.96	4.89	12.34	13.58	14.05	14.55	6.35	-2.73	-4.90	-1.55	0.32	1.84
0.96	0.89	0.73	0.70	2.42	8.06	10.99	[illegible]	9.33	3.04	-1.40	-5.65	-5.52	-1.27	0.71
0.80	0.69	0.46	0.14	0.95	3.91	5.93	5.73	3.68	0.26	-0.96	-3.85	-6.89	-2.84	0.06
0.64	0.42	0.22	-0.02	-0.03	0.66	2.23	2.33	1.08	0.44	-0.30	-1.20	-6.03	-5.41	-0.51
0.47	0.17	-0.03	-0.13	-0.12	-0.14	0.54	0.74	1.10	0.88	-0.14	-0.98	-3.70	-4.54	-1.60
0.29	-0.00	-0.26	-0.25	-0.06	0.05	0.51	0.95	1.02	0.84	-0.10	-0.56	-2.79	-3.25	-1.68
0.16	-0.15	-0.38	-0.30	0.03	0.36	0.71	1.06	0.78	0.45	-0.50	-0.47	-1.87	-2.82	-1.71
0.09	-0.23	-0.41	-0.27	0.15	0.53	0.72	0.85	0.36	-0.09	[illegible]	[illegible]	-0.39	-1.98	-1.86
0.04	-0.26	-0.41	-0.21	0.22	0.58	0.60	0.52	-0.03	[illegible]	[illegible]	[illegible]	0.11	-0.70	-1.43
-0.01	-0.29	-0.40	-0.17	0.26	0.52	0.45	0.24	-0.17	[illegible]	[illegible]	0.18	0.14	-0.20	-0.92
-0.07	-0.31	-0.35	-0.12	0.27	0.42	0.27	0.07	[illegible]	[illegible]	0.03	[illegible]	-0.01	-0.24	-0.58
-0.13	-0.33	-0.33	-0.07	0.23	0.30	0.14	[illegible]	[illegible]	[illegible]	0.07	[illegible]	0.16	-0.31	-0.42
-0.18	-0.35	-0.30	-0.06	0.18	0.19	[illegible]	[illegible]	[illegible]	0.09	[illegible]	[illegible]	0.15	-0.21	-0.31
-0.21	-0.35	-0.28	-0.07	0.11	[illegible]	[illegible]	[illegible]	0.01	[illegible]	[illegible]	[illegible]	-0.06	-0.05	-0.12
-0.22	-0.34	-0.28	-0.09	[illegible]	[illegible]	[illegible]	-0.04	[illegible]	[illegible]	[illegible]	0.03	0.01	-0.01	-0.06
-0.22	-0.33	-0.20	-0.12	[illegible]	[illegible]	-0.04	[illegible]	[illegible]	[illegible]	-0.01.	0.02	-0.03	-0.12	-0.16
-0.24	-0.35	-0.29	-0.14	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	-0.05	-0.01	-0.01	-0.11	-0.25	-0.27
-0.28	-0.37	-0.32	-0.17	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]	-0.03	-0.00	-0.04	-0.16	-0.28	-0.26
-0.33	-0.40	-0.35	-0.20	-0.09	[illegible]	[illegible]	[illegible]	-0.04	-0.03	-0.04	-0.08	-0.15	-0.18	-0.12
-0.37	-0.43	-0.36	-0.20	[illegible]	[illegible]	[illegible]	-0.09	-0.05	-0.04	-0.09	-0.12	-0.10	-0.04	0.05
-0.39	-0.43	-0.36	-0.18	[illegible]	[illegible]	-0.12	-0.10	-0.05	-0.07	-0.11	-0.12	-0.04	0.09	0.14
-0.39	-0.42	-0.34	-0.17	[illegible]	-0.07	-0.11	-0.10	-0.06	-0.07	-0.12	-0.10	0.01	0.13	0.11
-0.37	-0.40	-0.31	-0.15	-0.06	-0.07	-0.11	-0.10	-0.07	-0.08	-0.11	-0.08	0.01	0.08	0.02
-0.35	-0.38	-0.29	-0.15	-0.07	-0.08	-0.11	-0.09	-0.08	-0.08	-0.10	-0.09	-0.03	-0.03	-0.08
-0.33	-0.35	-0.28	-0.16	-0.08	-0.09	-0.10	-0.09	-0.08	-0.07	-0.09	-0.10	-0.09	-0.11	-0.12
-0.31	-0.34	-0.27	-0.16	-0.11	-0.10	-0.10	-0.11	-0.09	-0.07	-0.10	-0.13	-0.13	-0.14	-0.12
-0.31	-0.33	-0.27	-0.17	-0.12	-0.10	-0.11	-0.12	-0.10	-0.08	-0.12	-0.15	-0.13	-0.11	-0.09

Strong Updraft

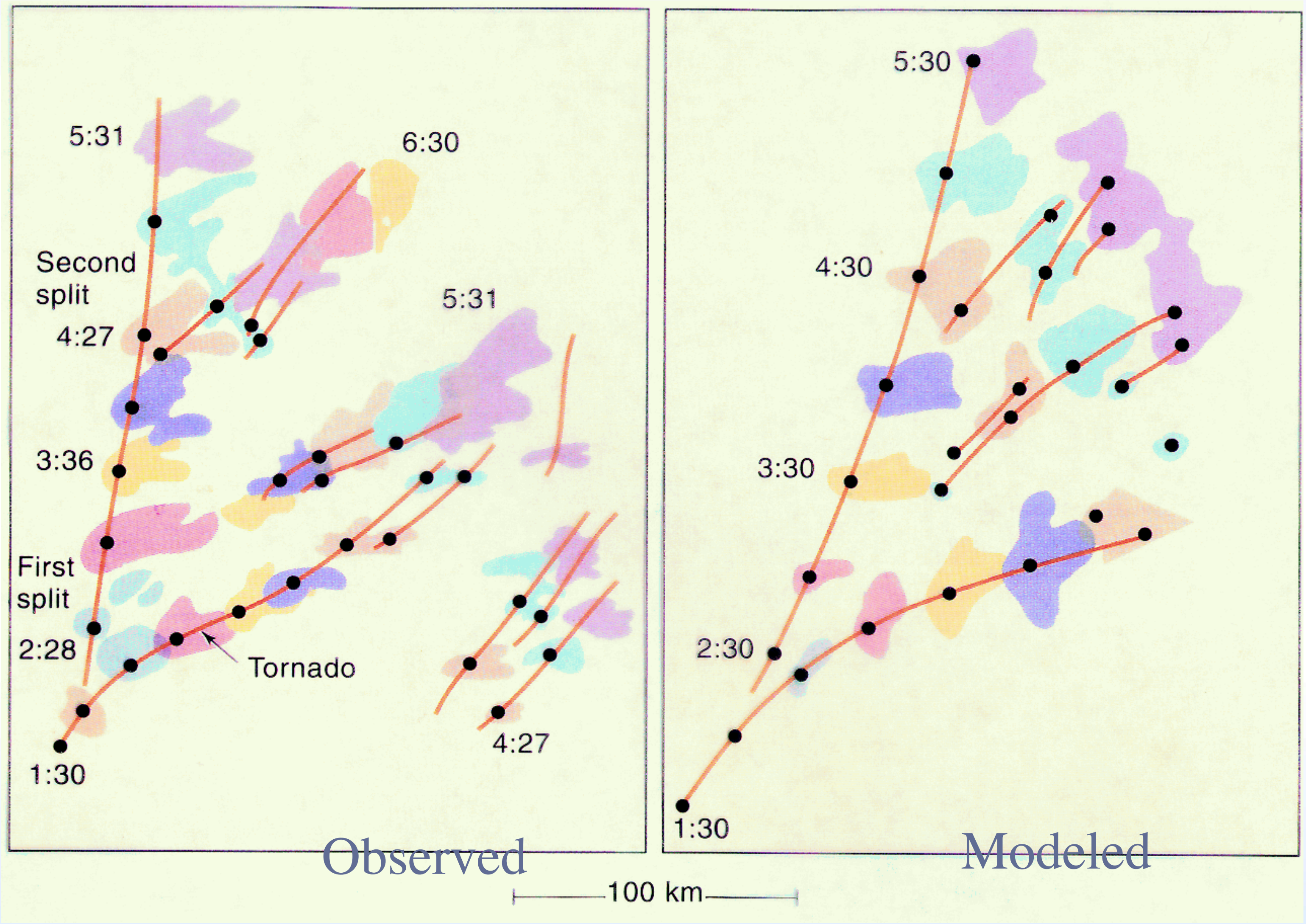
Heroic Analysis and Visualization!


Better Than Reading Numbers
Printer Contouring
Here's That Strong Updraft

Animation in Its Infancy

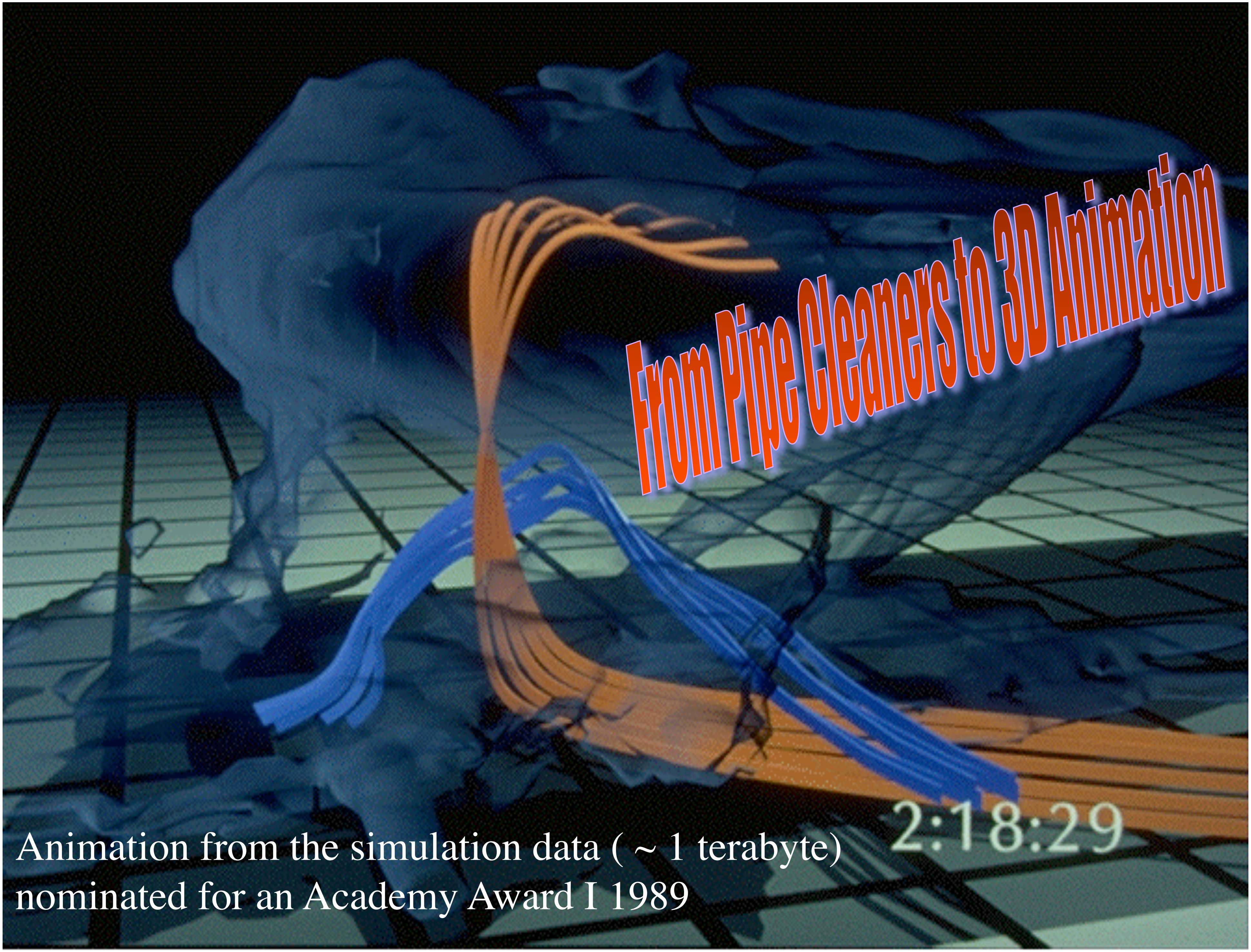


Supercell and Splits

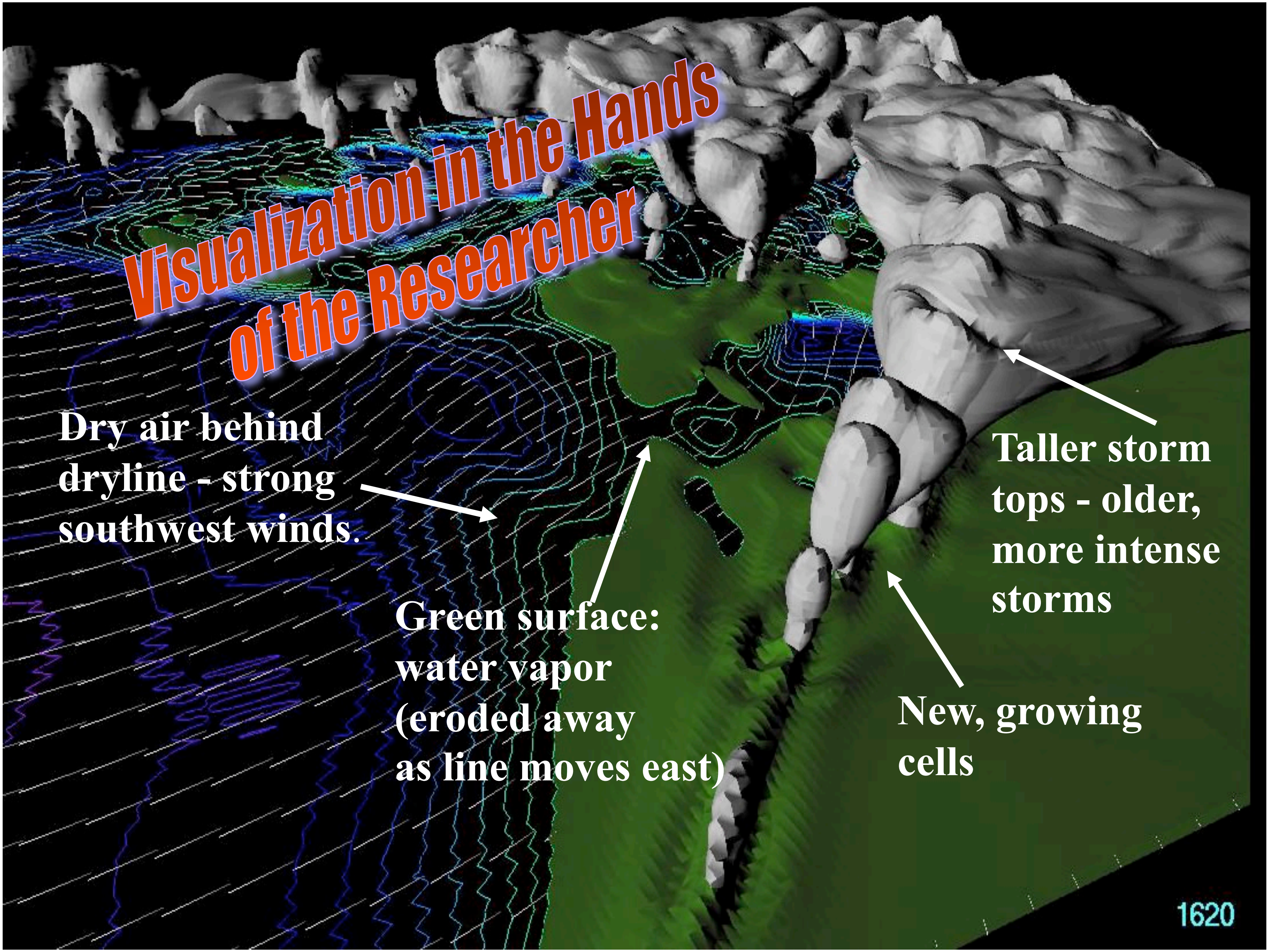


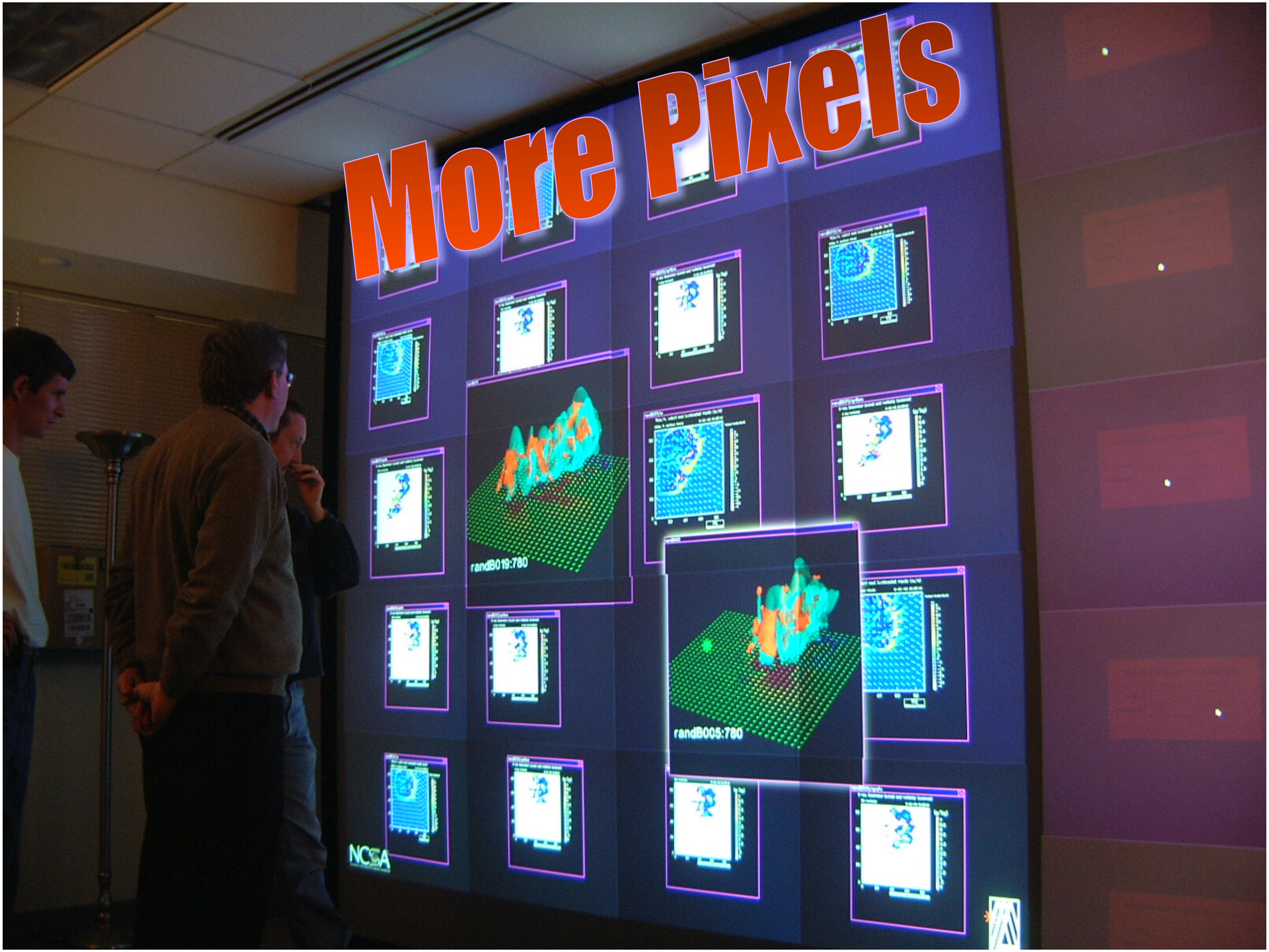
Pipe Cleaners Reveal Updraft/Downdraft Coupling in a Supercell

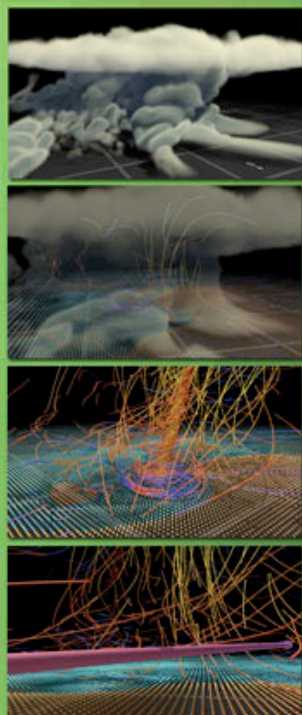
From Pipe Cleaners to 3D Animation



Animation from the simulation data (~ 1 terabyte)
nominated for an Academy Award I 1989


Visualization in the Hands
of the Researcher
Dry air behind
dryline - strong
southwest winds.
Green surface:
water vapor
(eroded away
as line moves east)
Taller storm
tops - older,
more intense
storms
New, growing
cells
1620


More Pixels
randB019:780
randB005:780
NCSA

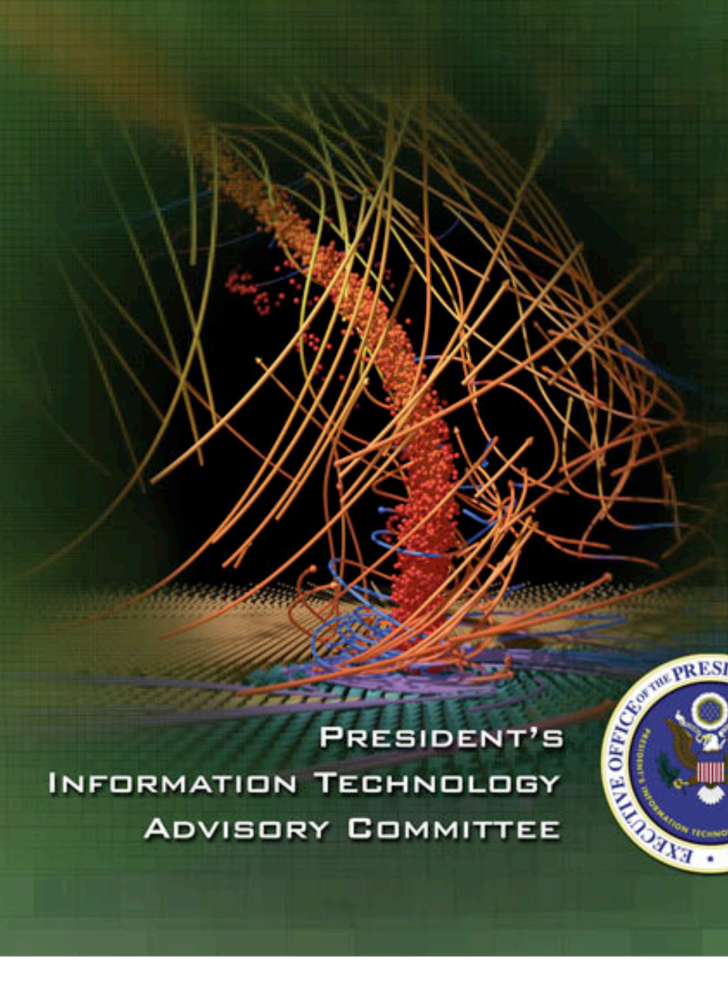


NATIONAL COORDINATION OFFICE
FOR INFORMATION TECHNOLOGY
RESEARCH AND DEVELOPMENT
SUITE II-405
4201 WILSON BOULEVARD
ARLINGTON, VIRGINIA 22230
(703) 292-4873
EMAIL ADDRESS: NCO@NITRD.GOV
WEB ADDRESS: HTTP://WWW.NITRD.GOV/PITAC

REPORT TO THE PRESIDENT

JUNE 2005

COMPUTATIONAL SCIENCE: ENSURING AMERICA'S COMPETITIVENESS

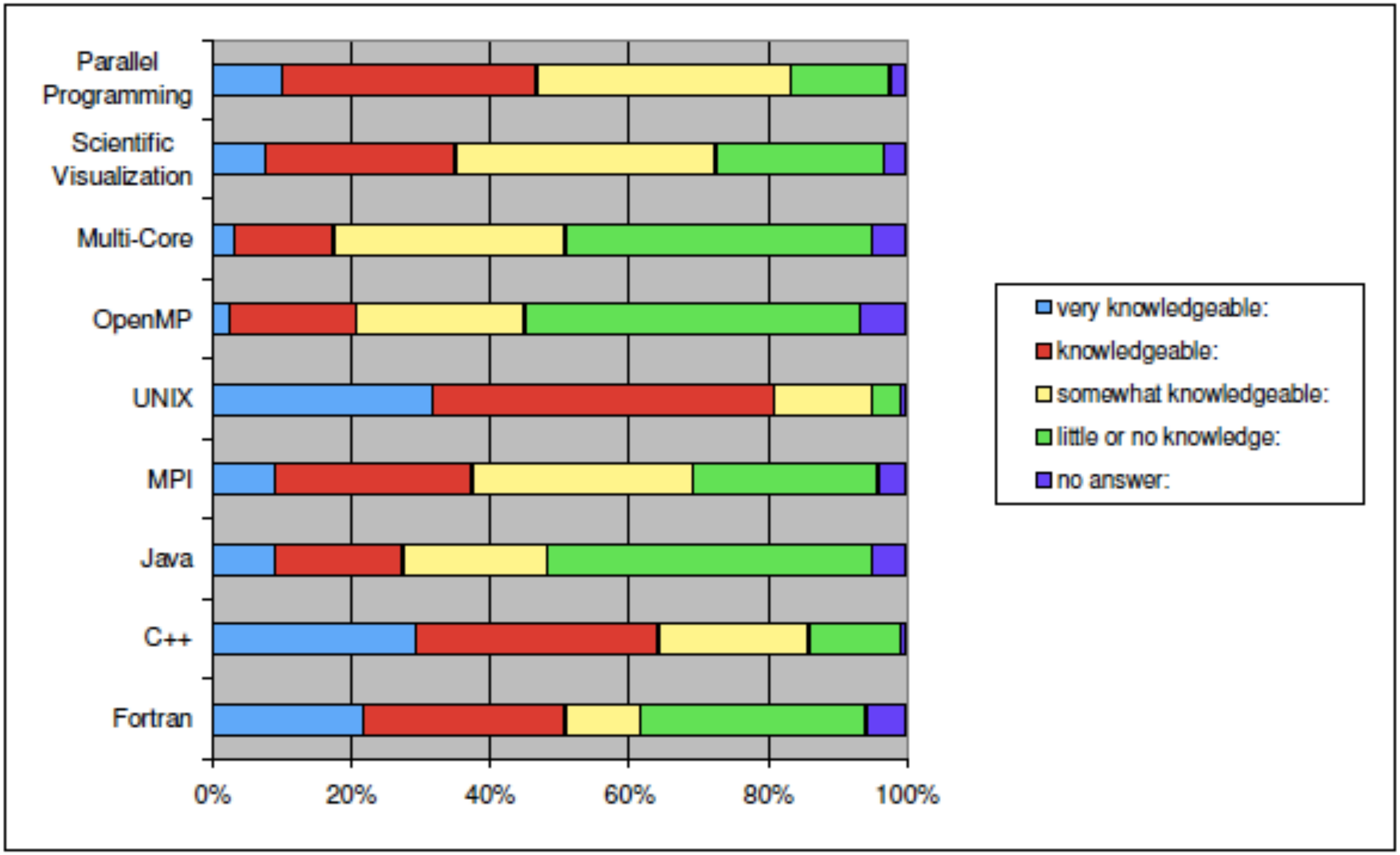


PRESIDENT'S
INFORMATION TECHNOLOGY
ADVISORY COMMITTEE

WHO ARE YOU (Departments)?

Astronomy	15
Geophysical Sciences	3
Chemistry	8
Computer Science	14
Applied Math	6
Physics	8
Biology	5
Engineering	42

WHAT DO YOU KNOW?



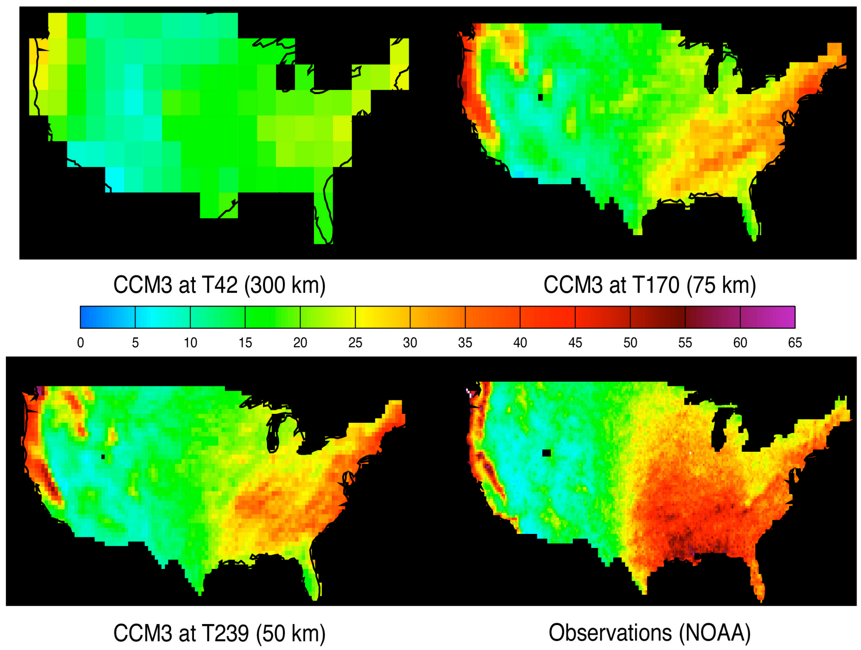
HPC Continues to Enable New Discovery

- Simulation of hurricanes
- Simulation of global climate change
- Simulation of molecular dynamics
- Simulation of hypersonic turbulence

Addressing Complexity

- Today's grand challenge problems often involve
  - Higher resolution
  - Use a variety of physics packages
  - Involve coupling of models (e.g. climate)

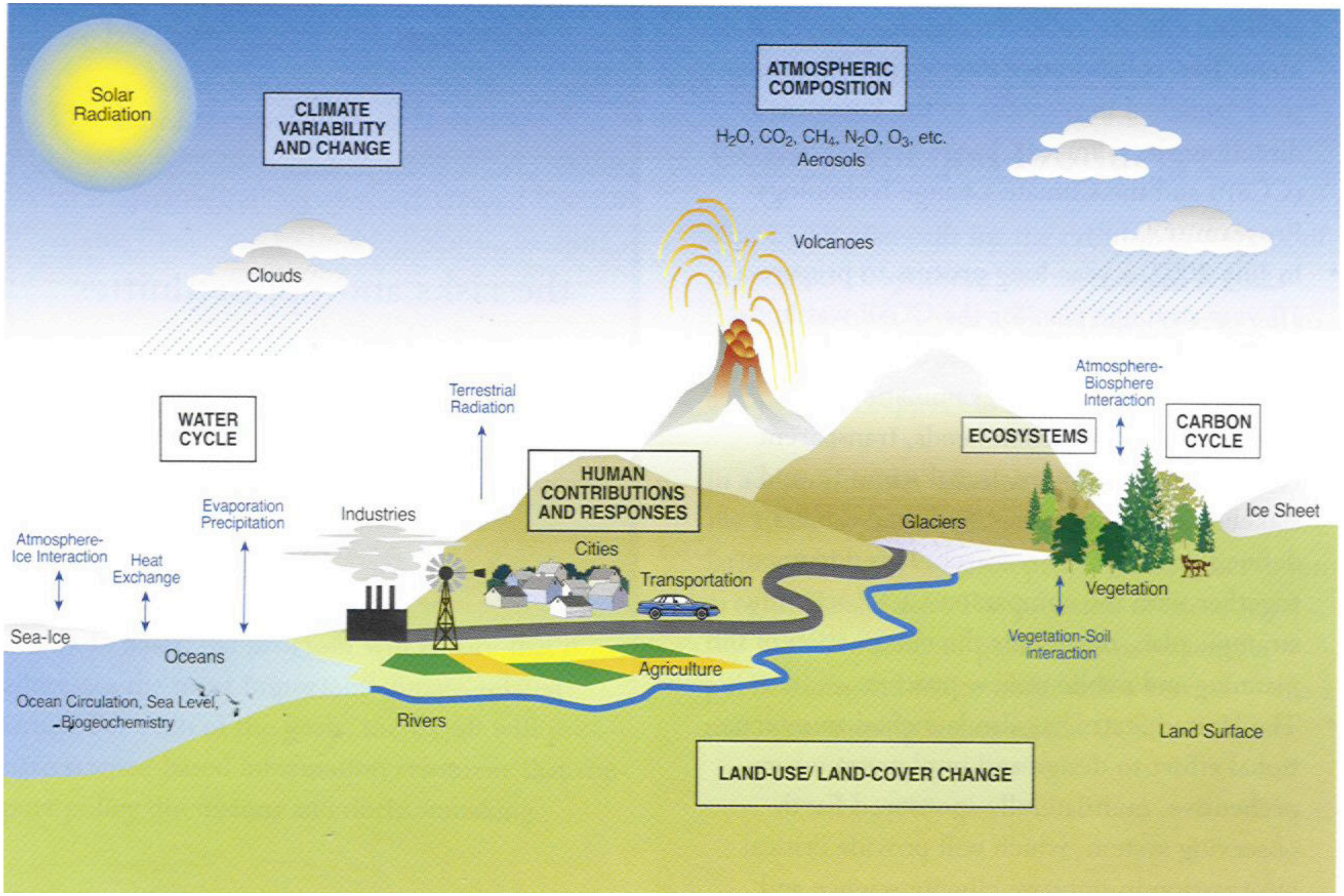
Model Resolution Influences Precipitation



CCM3 extreme precipitation events depend on model resolution. Here we are using as a measure of extreme precipitation events the 99th percentile daily precipitation amount. Increasing resolution helps the CCM3 reproduce this measure of extreme daily precipitation events.

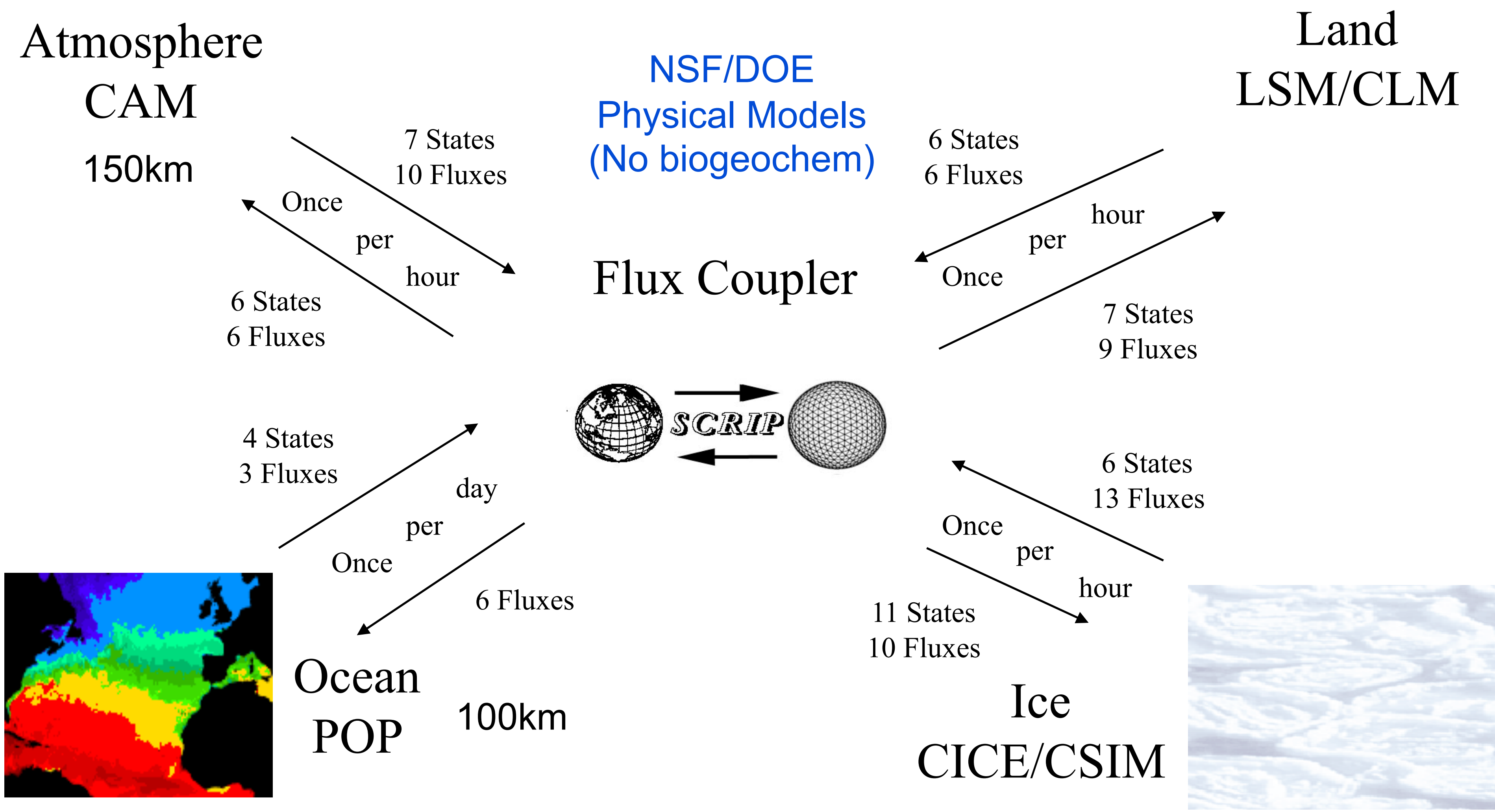
Source: Phil Jones

Climate and Complexity: Multiple Physics



Source: Phil Jones

Modeling Climate Complexity:
Multiple Models



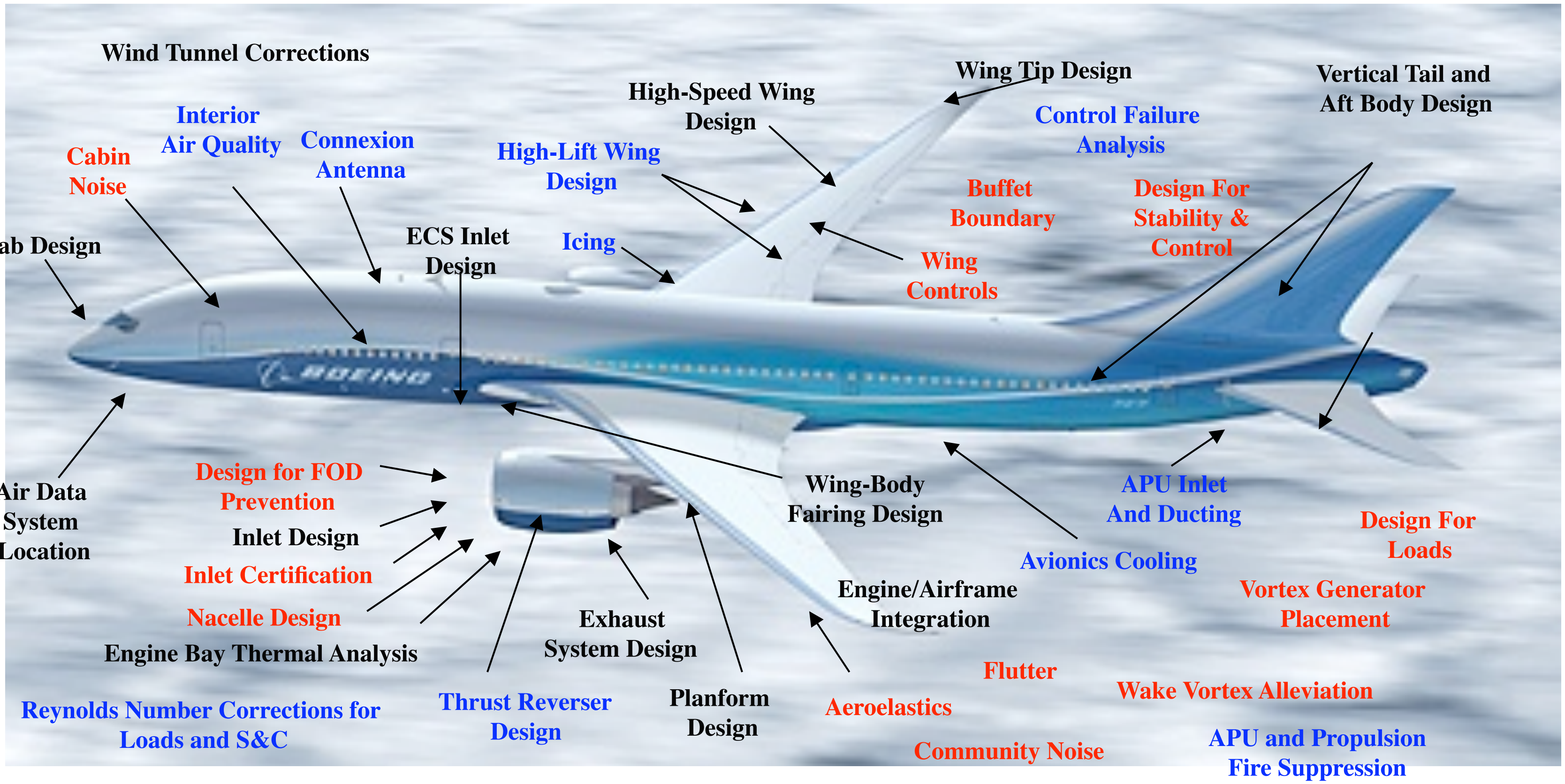
Source: Phil Jones

Boeing Computational Fluid Dynamics Penetration: Multiple Models

Much CFD
Opportunities: higher accuracy and expanded complexity

Some CFD
Opportunities: significant increases in design process speed and application

CFD opportunity

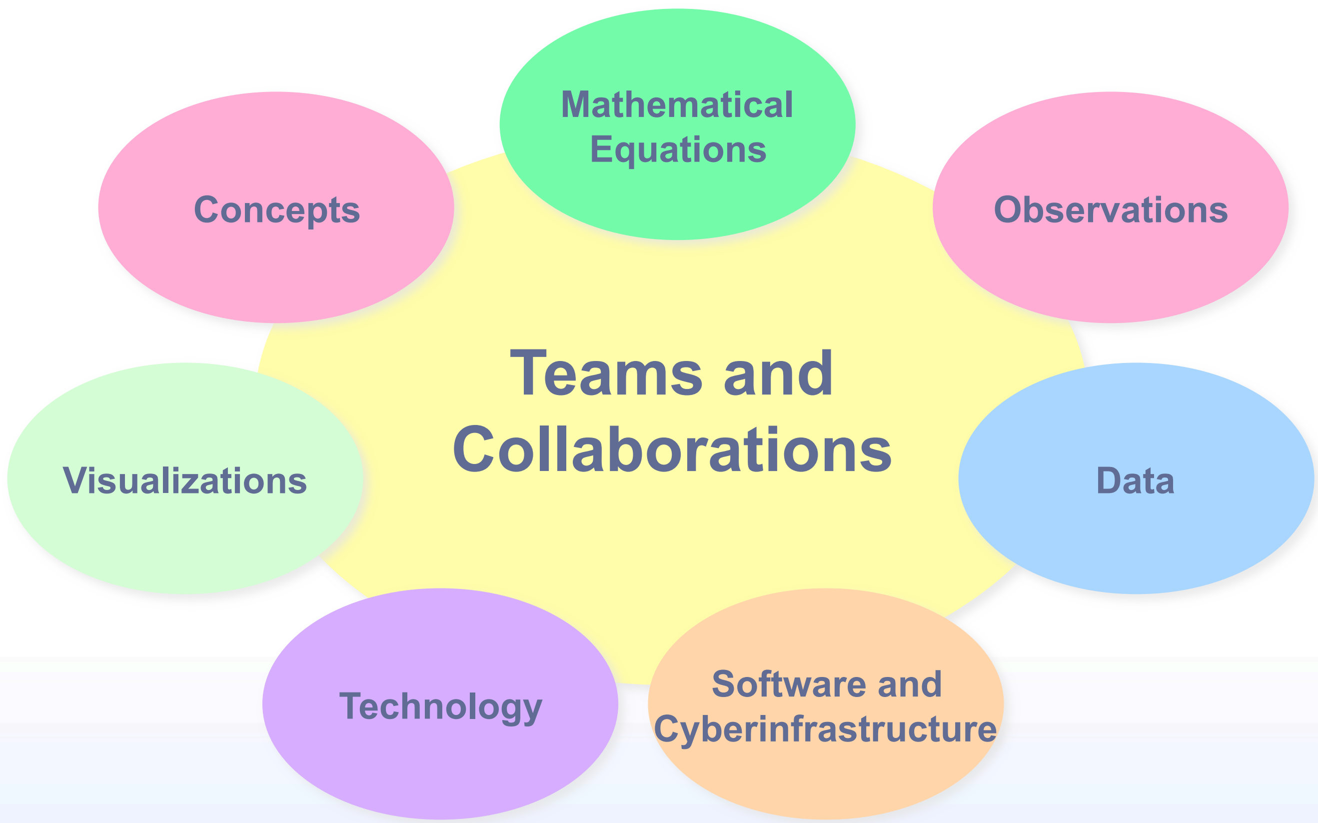


Thanks to HPC, virtual prototyping was used to develop the 787 Dreamliner. Boeing conducted tens of thousands of virtual wing prototypes, yet only 11 physical wind tunnel tests.
Image courtesy The Boeing Company.

Addressing Complexity

- Today's grand challenge problems often involve
  - Higher resolution
  - Use a variety of physics packages
  - Involve coupling of models (e.g. climate)
- Solving these problems typically requires teams with expertise in science, computational science, computer science

Today's Scientist, Researcher, or Student's Ecosystem

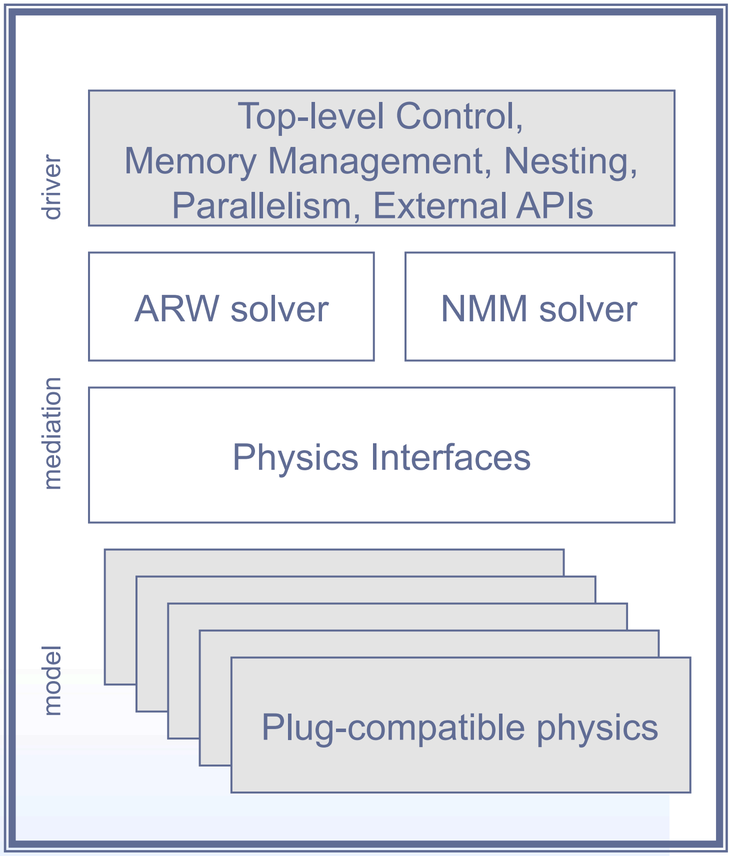


Addressing Complexity

- Today's grand challenge problems often involve
  - Higher resolution
  - Use a variety of physics packages
  - Involve coupling of models (e.g. climate)
- Solving these problems typically requires teams with expertise in science, computational science, computer science
- Model development for solving these problems
  - Takes years
  - Involves community contributions to the code
  - Involves development/use of simulation frameworks to remove the computational and workflow complexities as much as possible from the purview of the researcher

Weather Research and Forecast Model

- Large collaborative effort to develop next-generation community model with direct path to operations
- Advanced Software Architecture
  - Modular, flexible, extensible
  - Portable and efficient
  - Designed for HPC
- Applications
  - Atmospheric Research
  - Numerical Weather Prediction
  - Coupled modeling systems
  - Air quality research/prediction
  - High resolution regional climate
- 4000+ registered users
  - Operations, research (weather and regional climate), education, operations



WRF Supported Platforms

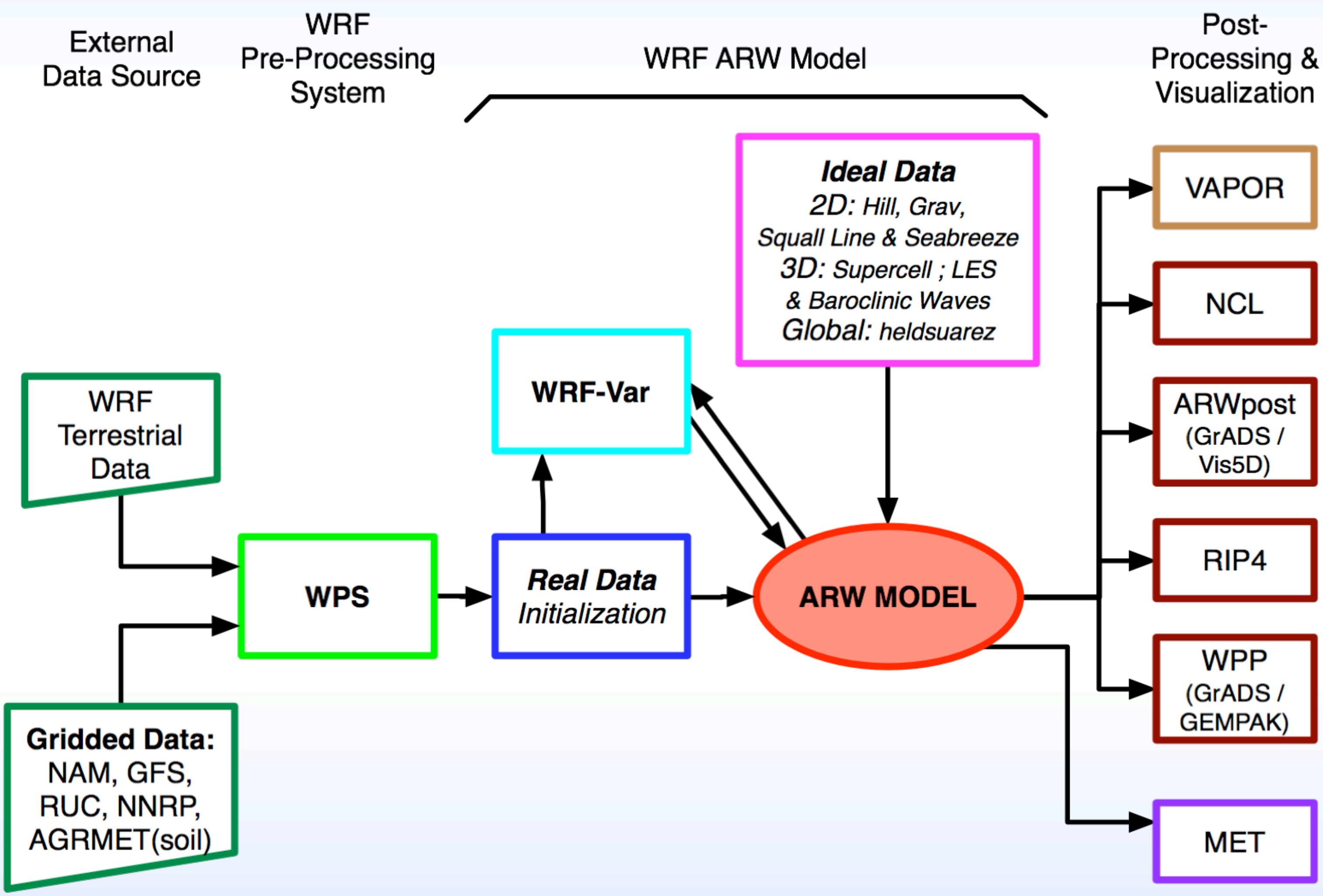
Vendor	Hardware	OS	Compiler
Apple	G5	MacOS	IBM
Cray Inc.	X1, X1e	UNICOS	Cray
	XT3/XT4 (Opteron)	Linux	PGI
HP/Compaq	Alpha	Tru64	Compaq
	Itanium-2	Linux	Intel
		HPUX	HP
IBM	Power-3/4/5/5+	AIX	IBM
	Blue Gene/L	Linux	IBM
	Opteron		Pathscale, PGI
NEC	SX-series	Unix	Vendor
SGI	Itanium-2	Linux	Intel
	MIPS	IRIX	SGI
Sun	UltraSPARC	Solaris	Sun
various	Xeon and Athlon	Linux and Windows CCS	Intel, PGI
	Itanium-2 and Opteron		

Petascale precursor systems

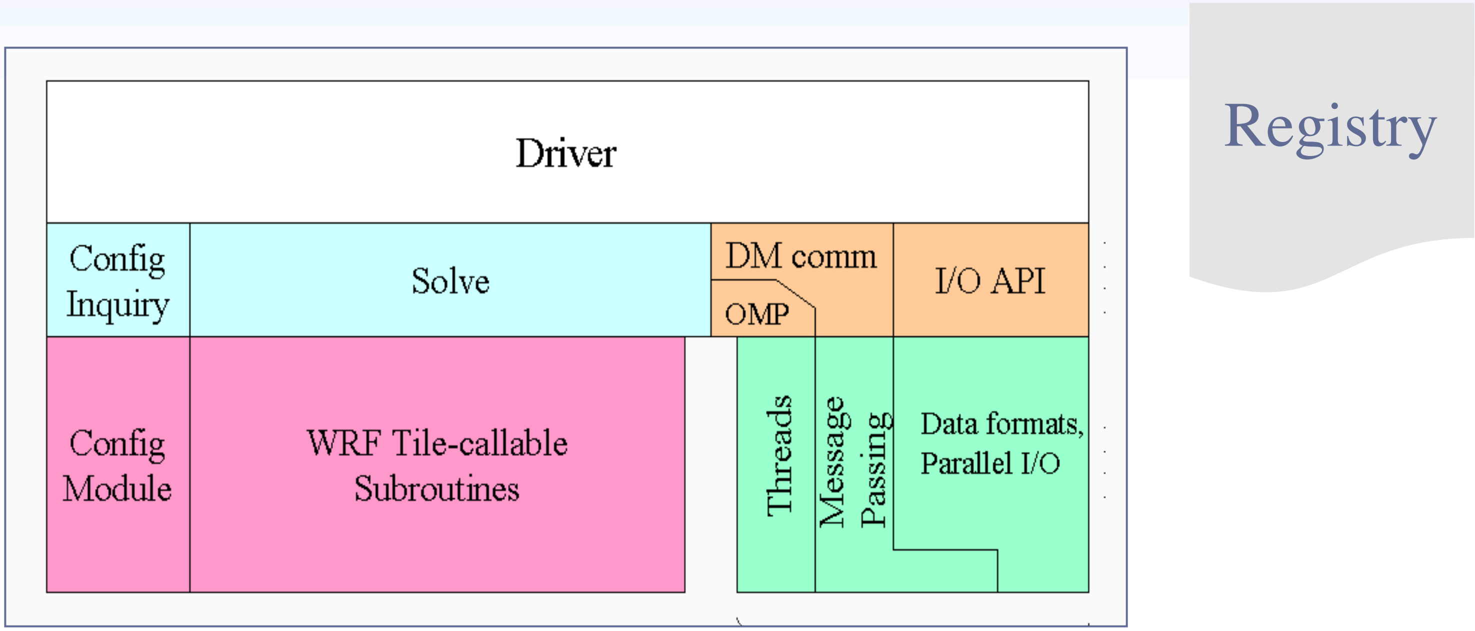
Source: John Michalakes, NCAR

NCSA

WRF ARW Modeling System Flow Chart



WRF Software Architecture



- **Hierarchical software architecture**
  - Insulate scientists' code from parallelism and other architecture/ implementation-specific details
  - Well-defined interfaces between layers, and external packages for communications, I/O, and model coupling facilitates code reuse and exploiting of community infrastructure, e.g. ESMF.

WRF Software Architecture

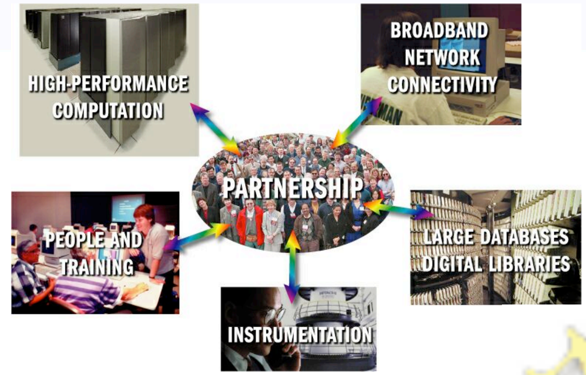
- **Driver Layer**
  - **Allocates, stores, decomposes model domains, represented abstractly as single data objects**
  - Contains top-level time loop and algorithms for integration over nest hierarchy
  - Contains the calls to I/O, nest forcing and feedback routines supplied by the Mediation Layer
  - Provides top-level, non package-specific access to communications, I/O, etc.
  - Provides some utilities, for example **module_wrf_error**, which is used for diagnostic prints and error stops
- **Mediation Layer**
  - Provides to the Driver layer
    - Solve solve routine, which takes a domain object and advances it one time step
    - I/O routines that Driver calls when it is time to do some input or output operation on a domain
    - Nest forcing and feedback routines
    - The Mediation Layer and not the Driver knows the specifics of what needs to be done
  - The sequence of calls to Model Layer routines for doing a time-step is known in Solve routine
  - Responsible for dereferencing driver layer data objects so that individual fields can be passed to Model layer Subroutines
  - **Calls to message-passing are contained here**

WRF Software Architecture

- **Model Layer**
  - **Contains the information about the model itself, with machine architecture and implementation aspects abstracted out and moved into layers above**
  - **Contains the actual WRF model routines that are written to perform some computation over an arbitrarily sized/shaped subdomain**
  - All state data objects are simple types, passed in through argument list
  - Model Layer routines don't know anything about communication or I/O; and they are designed to be executed safely on **one thread** – they <u>never</u> contain a **PRINT**, **WRITE**, or **STOP** statement
  - These are written to conform to the Model Layer Subroutine Interface (more later) which makes them "tile-callable"
- **Registry: an "Active" data dictionary**
  - Tabular listing of model state and attributes
  - Large sections of interface code generated automatically
  - Scientists manipulate model state simply by modifying Registry, without further knowledge of code mechanics

Complexity, Cyberinfrastructure, and HPC

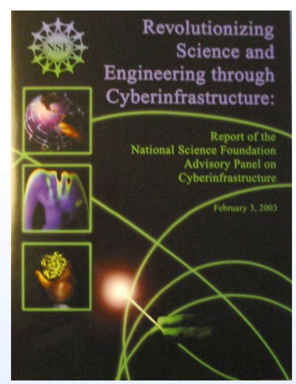
Cyberinfrastructure is the coordinated aggregate of software, hardware and other technologies, as well as human expertise, required to support current and future discoveries in science and engineering.



National Science Foundation's Cyberinfrastructure

"Thanks to Cyberinfrastructure **and information systems, today's scientific tool kit includes distributed systems of hardware, software, databases and expertise that can be accessed in person or remotely."**

Arden Bement, NSF Director
February, 2005



NSF Blue Ribbon Panel (Atkins) Report provided compelling and comprehensive vision of an integrated Cyberinfrastructure

LEAD CyberInfrastructure (A Team Effort)

A Cyberinfrastucture for Mesoscale Meteorology Research, Forecasting, and Education Involving Meteorologists and Applications and Computer Scientists

https://portal.leadproject.org/

The University of Oklahoma
University Corporation for Atmospheric Research
Colorado State University
Howard University
Millersville University

LINKED ENVIRONMENTS FOR ATMOSPHERIC DISCOVERY

University of Illinois at Urbana-Champaign
Indiana University — Quality Education, Lifetime Opportunities.®
UNC Chapel Hill
UAH — The University of Alabama in Huntsville

LEAD and a New Level of Complexity

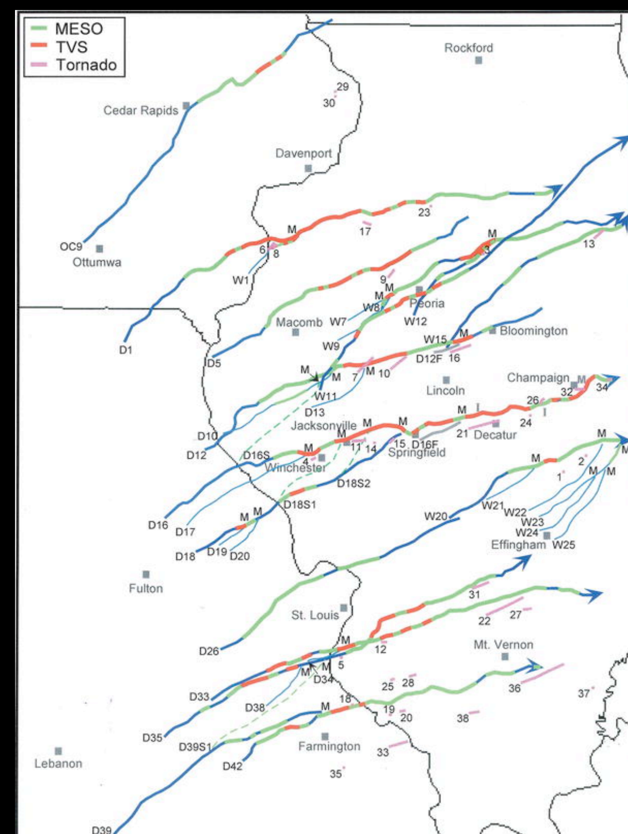
LEAD was funded to develop a comprehensive national cyberinfrastructure for mesoscale meteorology research, education, and prediction. It is addressing the fundamental information technology (IT) research challenges needed to **create an integrated, scalable environment for**

identifying,
accessing,
preparing,
assimilating,
predicting (WRF)
managing,
analyzing,
mining, and
visualizing

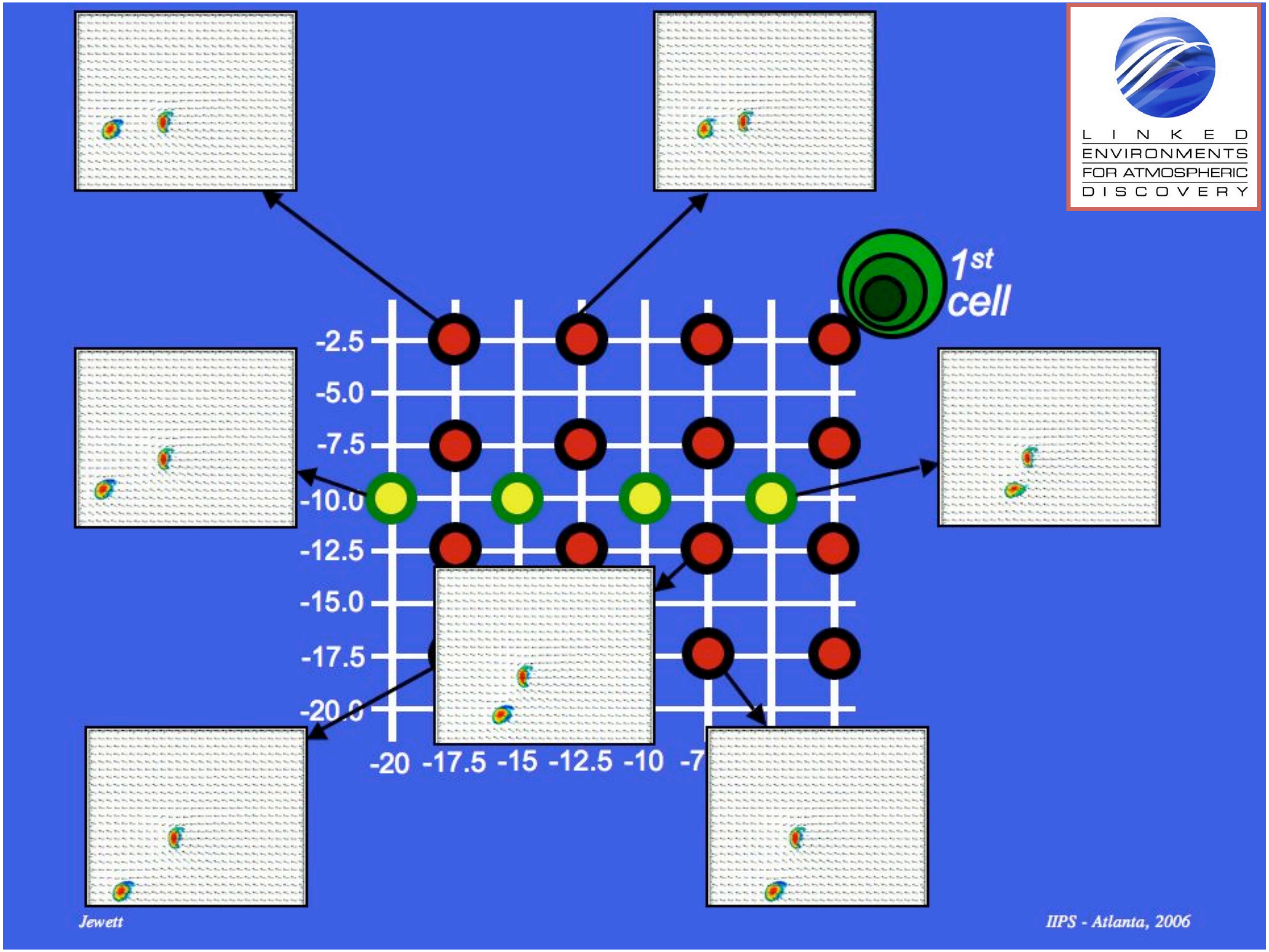
a broad array of meteorological data and model output, independent of format and physical location and having *dynamically adaptive, on-demand* response.

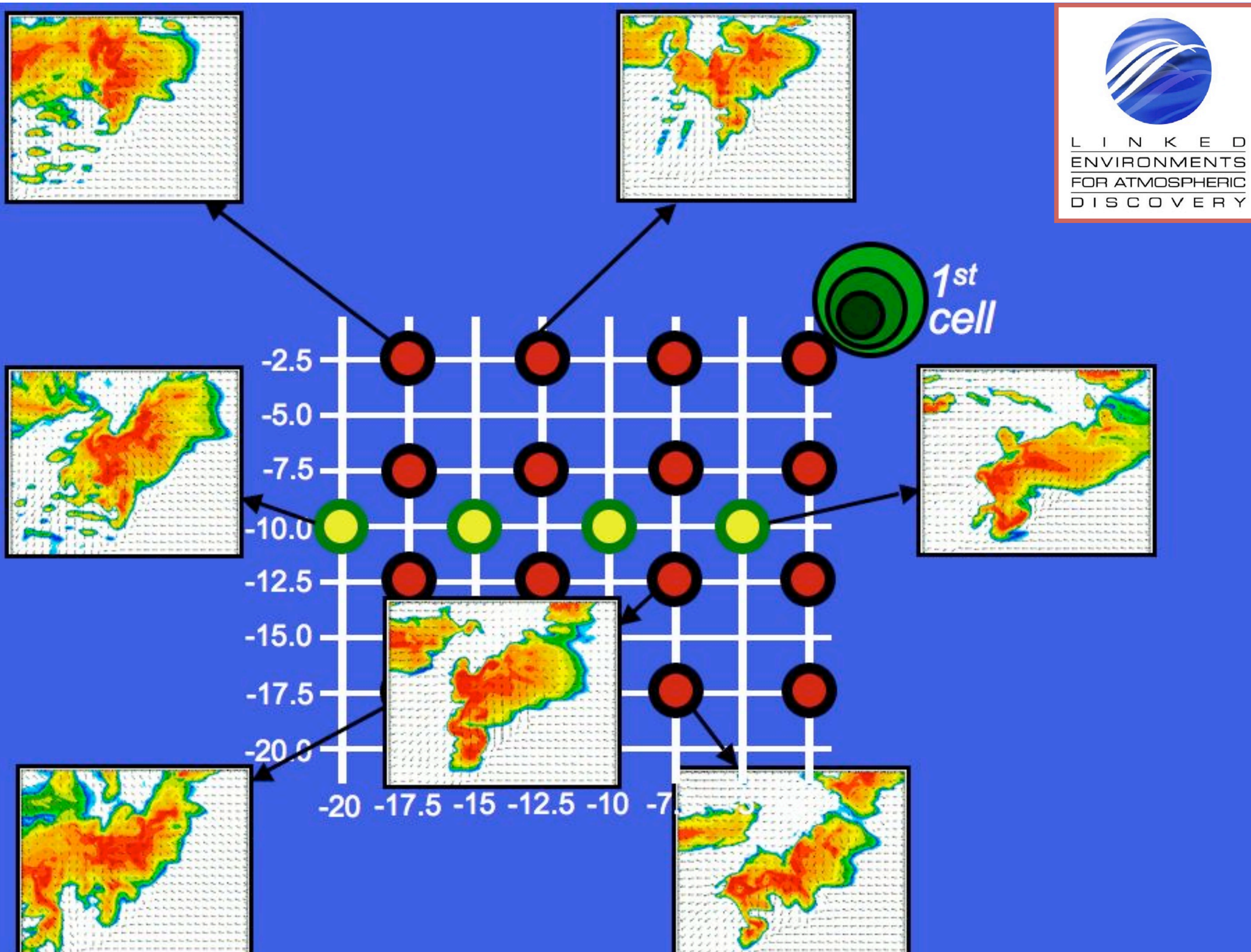
April 1996 Illinois Tornadoes

Storm interaction - a focus of Project VORTEX-II (2009)



Source: Brian Jewett


LINKED
ENVIRONMENTS
FOR ATMOSPHERIC
DISCOVERY
1st cell
-2.5
-5.0
-7.5
-10.0
-12.5
-15.0
-17.5
-20.0
-20 -17.5 -15 -12.5 -10 -7
Jewett
IIPS - Atlanta, 2006


LINKED ENVIRONMENTS FOR ATMOSPHERIC DISCOVERY
1st cell
-2.5
-5.0
-7.5
-10.0
-12.5
-15.0
-17.5
-20.0
-20
-17.5
-15
-12.5
-10

Automatically Triggered Forecasts

http://banff.atmos.uiuc.edu/trigger/

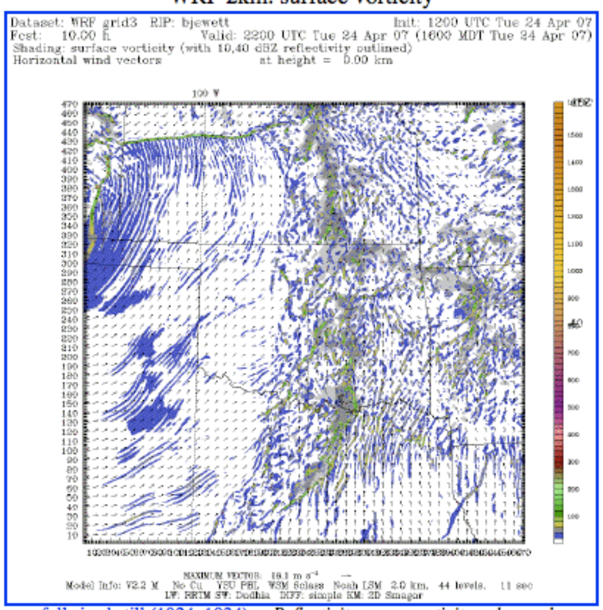
2-km WRF triggered forecasts (WPS/WRF) at NCSA

This particular WRF forecast was made April 24, 2007. Click on either image for an animation.

- Nested, 18/6/2km
- NAM/WPS initialization
- Data saved every 10 min
- Forecast carried out on tungsten
- Triggered run (SPC MDs/Watches), executed via NCSA ensemble broker

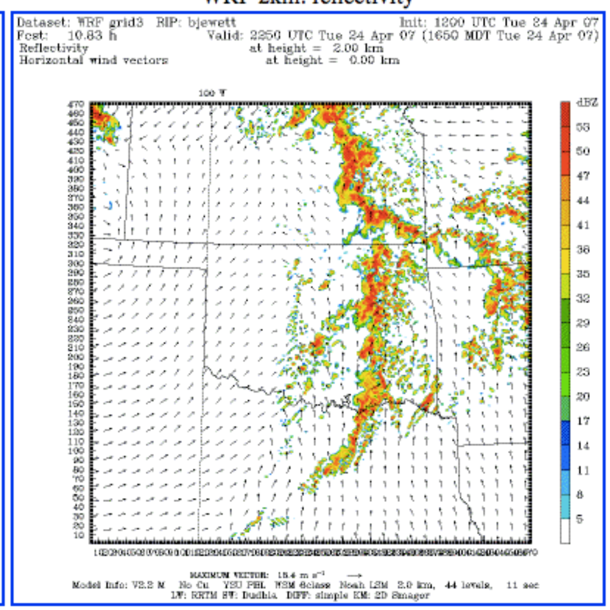
  Run archived on Unitree at shawn/Broker/200704241819-000000.35.48-97.46
- Batch job 947158 (tungsten?); init 12z, nest started 18z;
  center lat/lon 35.47591228070176 -97.4601052631579
- 2-D images: 800x800, 2048x2048, NCAR metacode (view latter with idt via X11)
- 3-D images are here

WRF 2km: surface vorticity

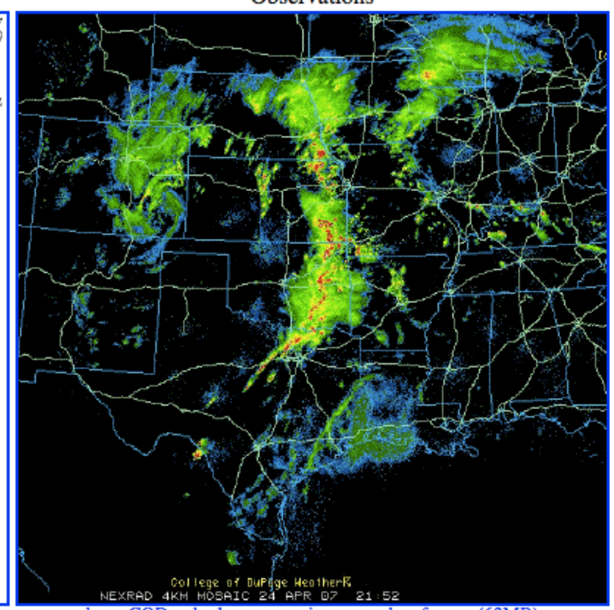


full sized still (1024x1024) *Reflectivity gray; vorticity color scale*

WRF 2km: reflectivity



Observations

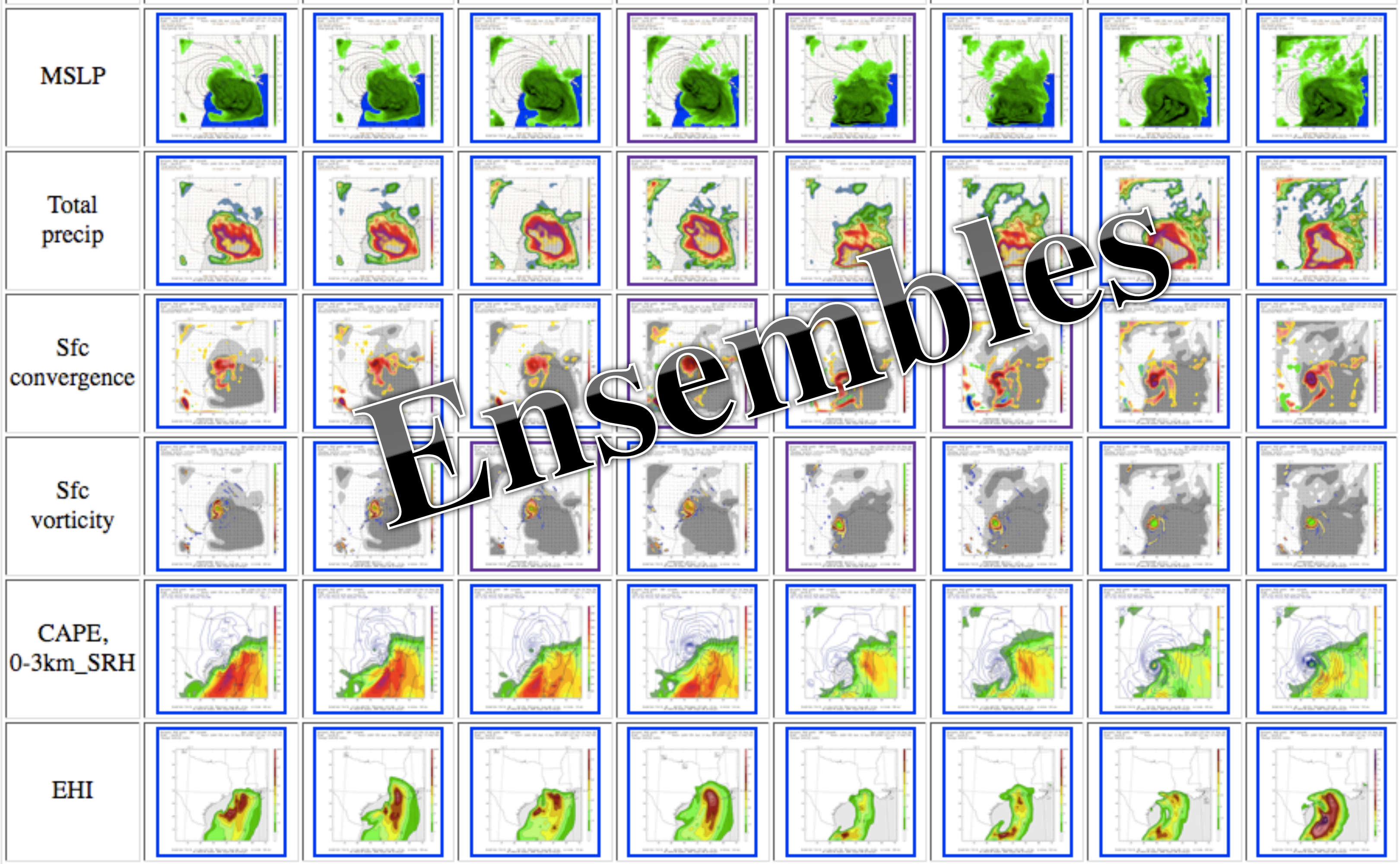


large COD radar loop, every image, no key frames (62MB)

WRF 8-member ensemble forecast Fri Sep 12, 2008 07:19 **Time = 24.0 Hours**

WRF-ARW ***Select time (hours):*** *0 6.0 12.0 18.0 24.0* ***All*** Click on any image icon for a loop of all available times.

Member	GFS-A	GFS-B	GFS-C	GFS-D	NAM-A	NAM-B	NAM-C	NAM-D
Run	18 km, 24 hr	18 km, 24 hr	18 km, 24 hr	18 km, 24 hr	18 km, 24 hr	18 km, 24 hr	18 km, 24 hr	18 km, 24 hr
Microphys	WSM-3	WSM-3	WSM-3	WSM-3	WSM-3	WSM-3	WSM-3	WSM-3
Sfc phys/PBL	Noah_LSM, YSU	RUC_LSM, MYJ	Noah_LSM, YSU	RUC_LSM, MYJ	Noah_LSM, YSU	RUC_LSM, MYJ	Noah_LSM, YSU	RUC_LSM, MYJ
Cumulus	GrellV2	GrellV2	Kain-Fritsch	Kain-Fritsch	GrellV2	GrellV2	Kain-Fritsch	Kain-Fritsch
IC data	GFS	GFS	GFS	GFS	NAM	NAM	NAM	NAM



Week in Review – Opportunities Abound

- Use of multicore technology and accelerators form the basis for most petascale computing over the next decade
- Core counts today on the largest systems exceed 50,000
- Cache friendly codes will perform best on most systems – i.e. there are many flops per memory fetch
- Hybrid programming (e.g. using OpenMP on the SMP and MPI across SMPs) may boost performance for some applications
- Fault – tolerance: remember to checkpoint your data or implement a fault tolerance schema
- Analysis and visualization may need to be done inline – as a simulation proceeds – if the data volume being produced is voluminous (petabytes)
- Document your code
- Instrument your code for debugging and performance analysis

Why and When to Use Charm++ and AMPI

- When you need automatic dynamic load balancing
- Automatic overlap of computation / communication
- Parallel Composition:
  - If you are composing multiple parallel modules
  - Charm++ can interleave their execution, overlapping idle time
- Automatic fault tolerance
  - Caveat: if the machine/scheduler doesn't kill a job when a node fails
- Mature, scalable system, with support for interactive debugging, live visualization, performance tools,

- Charm++: is C++ programming, but can interface with Fortran
- AMPI: C, C++, Fortran
  - Gets most benefits of Charm++ for MPI programs
  - Currently requires a small conversion effort on some machines (automated on many machines)

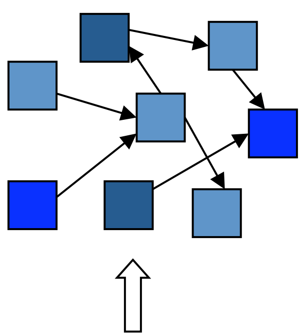
Migratable Objects (aka Processor Virtualization)

Programmer: **[Over] decomposition into virtual processors**

Runtime: **Assigns VPs to processors**

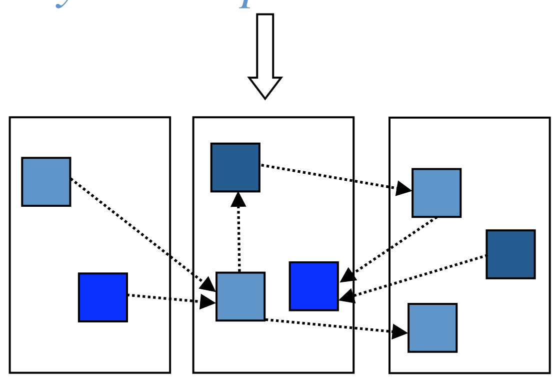
Enables *adaptive runtime strategies*

Implementations: Charm++, AMPI



User View

System implementation



Benefits

- Software engineering
  - Number of virtual processors can be independently controlled
  - Separate VPs for different modules
- Message driven execution
  - Adaptive overlap of communication
  - Predictability :
    - Automatic out-of-core
    - Prefetch to *local stores*
  - Asynchronous reductions
- Dynamic mapping
  - Heterogeneous clusters
    - Vacate, adjust to speed, share
  - Automatic checkpointing
  - Change set of processors used
  - Automatic dynamic load balancing
  - Communication optimization

Why Use Libraries: The Reality For DGEMM

- N=100
  - 1818 MF (1.1ms) – great performance compared to core peak performance
- N=1000
  - 335 MF (6s) – should be ~1 s based on core peak
- What this tells us:
  - Obvious expression of algorithms are not transformed into leading performance
  - Compilers do not magically solve many performance problems
  - We need to understand in detail the system architecture we are working on to write fast code (e.g. effectively use the cache and network structure of the system) to the degree we can control it

Faster (Better Algorithms) Often Available in Libraries

- Modern algorithms can provide significantly greater performance
- Example: Solving systems of linear equations
  - For most of the history of computing, as much of an improvement in performance in solving systems of linear equations arising from PDEs came from better algorithms as from faster hardware
  - From Gauss – Seidel to MultiGrid solvers

Higher Level Parallel I/O Libraries Are Important

- Without careful consideration, I/O can dominate your wall clock time
- Scientific applications may use significant I/O capabilities in preparing data for initializing a simulation, for checkpointing or other fault tolerant schema, and for writing out data to analyze and visualize later
- netCDF and HDF5 are two popular “higher level” I/O libraries
    - Abstract away details of file layout
    - Provide standard, portable file formats
    - Include metadata describing contents

NCSA

Blue Waters Partnership

http://www.ncsa.illinois.edu/BlueWaters/

BLUE WATERS
BREAKING THROUGH THE LIMITS

IBM

Credit: © *2007 JupiterImages Corporation*

GREAT LAKES CONSORTIUM
FOR PETASCALE COMPUTATION

Giga- → Tera- → Peta- (Bytes, Flop/s) → Exa- → Zetta-

Blue Waters Computing System

System Attributes	Blue Waters*
Vendor	IBM
Processor	IBM Power7
Sustained Performance (PF)	~1
SMP size	≥16
Number of Processor Cores (GB)	>200,000
Memory per core	≥2
Amount of Disk Storage (PB)	>10
Amount of Archival Storage (PB)	>500
External Bandwidth (Gbps)	100-400

* *Reference petascale computing system (no accelerators).*

Blue Waters Computing System

- Be capable of optimized simultaneous multithreading
- Be capable of vector multimedia extension with four or more floating-point operations per cycle.
- Feature multiple levels of cache (private L1 and L2 caches for each core and an L3 shared cache)
- Support 10 or more data streams
- Provide an integrated network interconnect with significantly reduced latency and increased bandwidth.
- Allow overlapping of computation with I/O and node communication through RDMA technology

Blue Waters Computing System

- ***System software:***
  - IBM's LoadLeveler for resource management
  - Blue Waters software will include Fortran, Co-Array Fortran, C/C++, and UPC compilers
  - GPFS file system and software

- ***Applications libraries:***
  - MASS, ESSL, and Parallel ESSL math libraries
  - MPI I/O
  - VisIt (parallel visualization)

Blue Waters Computing System

- ***Programming models and environments:***
  - MPI and MPI2
  - OpenMP for shared memory
  - Partitioned Global Address Space languages
  - Low-level active messaging layer
  - Debugging tools
  - HPC and HPCS Performance Toolkits
  - The CHARM++ and Cactus frameworks
  - Eclipse-based application framework to support development

Petascale Computing – Expertise needed in

- ***Application algorithms*: serial and parallel, non-numeric and numeric algorithms, libraries, shared memory techniques, hybrid method tuning, SIMD/vectorization.
- ***System hardware*: architecture, cache use, instruction-level parallelism, communication/synchronization topology, I/O.
- ***System software*: operating systems, compilers, performance tools, MPI/OpenMP, debuggers, job scheduling, file systems, archiving, system check pointing, dynamic reconfiguration, fault recovery, code development environments.
- ***Performance analysis*: understanding software/hardware interaction, identifying current performance bottlenecks and projecting future performance, optimizing and measuring performance.
- ***Modeling and projection:*** application and algorithm analysis and developing models of performance to explain current or future performance.
- ***Optimization and benchmarking:*** software development and *:*enhancement, measuring and verifying performance and correctness.
- ***Application simulation*** expertise in simulation methodology, including execution-driven, trace-driven, whole system, and network simulation.

Pathways to Blue Waters Workshop: *Communication Intensive Algorithms and Applications*

- Held at NCSA last October
- Attended by nearly 100 researchers from universities and other organizations
- Goals
  - To identify key challenges/issues in scaling applications efficiently to >200,000 cores
  - To learn about and discuss different communication fabrics and programming strategies for efficiently using them
  - To gain a better understanding of algorithm/application communication needs and to identify programming strategies for dealing with them
  - To identify I/O performance issues and potential solutions
  - To identify issues and associated strategies for debugging and running very large applications
  - To encourage communication and sharing among groups preparing for sustained petascale computing
- Answers to a questionnaire reveals a diversity of responses and the need to think carefully how to best modify or develop peta capable codes

What programming and scripting languages do you most commonly use?

- Fortran 77, 90, 95
- C/C++
- Perl
- Python C/C++
- Shell scripting
- APIs such as OpenMP and MPI
- Bash
- Super Instruction Assembly Language (SIAL)
- Ruby
- Tcl
- xml
- MATLAB and Star-P

What math or statistical library routines do you most commonly use?

- BLAS (DGEMM)
- FFTW, FFTW3 (3d FFT)
- PETSc
- Linear algebra, nonlinear equation solvers, differential-algebraic equation solvers
- appspack.
- Intel MKL
- LAPACK
- SCALAPACK
- ESSL
- PESSL
- GSL
- HYPRE
- NAG
- A++/P++
- Overture
- SPRNG Parallel Random Number Generator
- IBM optimized math intrinsic libraries (mass,massv)
- ARPACK
- MUMPS
- SuperLU
- PARPACK
- IMSL

What communication libraries do you most commonly use?

- MPI
- OpenMP
- MPI+
- SciDAC API library QMP
- GA
- ARMCI
- mixed-mode MPI +
- Pthreads
- Intels TBB
- Charm++
- PNNL Global Arrays (GA)

What performance tools do you most commonly use?

- RDTSC cycle counter instruction
- Tau
- MPITrace
- PAPI
- IPM
- cachegrind
- Vampir
- Vtune
- hpmcount
- PERI
- Gprof
- lcaperf
- Profiling tools from PETSc
- Hpm
- Shark
- Osiris has timing (MPI_Wtime) in the code
- MPIP
- Jumpshot
- CrayPAT

What debugging tools do you most commonly use?

- Objdump coupled with output from exceptions
- Printf
- TotalView
- gdb
- Visual studio for single processor
- dbx
- Eclipse
- valgrind
- cscope
- Objdump
- Super Instruction Processor (SIP) Tools
- DDT

Petascale Code Development

- Codes need to run on a variety of systems and therefore good code design and choice of algorithms is important (e.g. isolate parts of a code as much as possible that will require changes when moving from one system to another)
- Code test suites are important for validating model results
- Code writing skills need to be mastered over time
- Code development for petascale systems takes time – think and design before you write code
- Avoid barriers whenever you can
- Ensure that data is moved with large messages whenever possible
- Code debugging
  - be patient and get a good night's sleep if you are tired
  - Sometimes it helps to start with the code and check the algorithms by writing them down from the code
- Admonition: learn from those who have gone before – you are not alone!

Thanks to all who made this workshop possible!

Organizers

Support staff at all sites

Presenters

Attendees

- For coming
- For good questions
- For working hard at the hands-on exercises