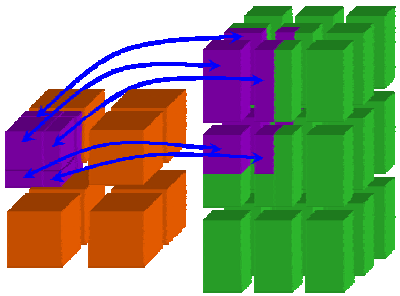
# MxN Data Redistribution
## for
# Coupling Disparate Parallel Components



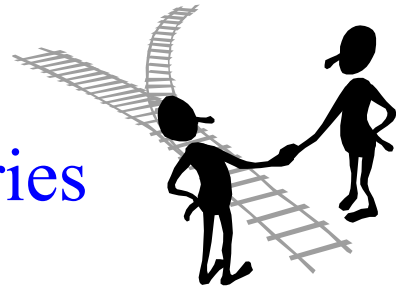
James A. Kohl
Computer Science and Mathematics Division
Oak Ridge National Laboratory

Interfaces in Parallel Multi-Physics Simulations (MS41)
SIAM PP04, Fisherman's Wharf
February 26, 2004

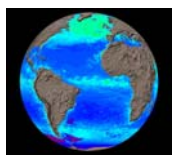
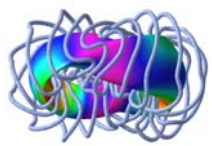


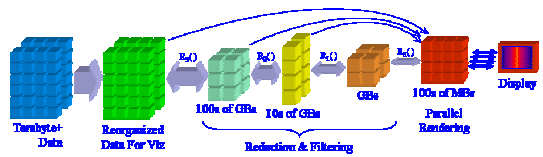
Research supported by the Mathematics, Information and Computational Sciences Office, Office of Advanced Scientific Computing Research, U.S. Department of Energy, under contract No. DE-AC05-00OR22725 with UT-Battelle, LLC.

# Coupled Multi-Physics Simulations



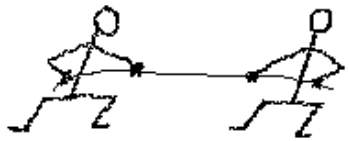
- Improved Fidelity from Live Coupling
  - ⇒ Dynamic Feedback versus Static Boundaries
  - ⇒ More Accurate Overall System Modeling
- Capitalize on Domain Expertise
  - ⇒ Combine "Best" Interdisciplinary Software
- Mechanism for Next-Generation Collaboration
- Applications of Coupling:
  - ⇒ Climate, Fusion, Biology, Numerical, Visualization

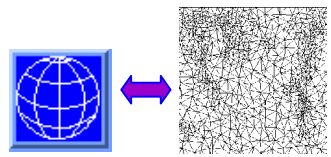



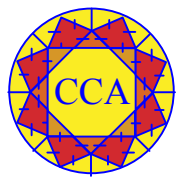
# What's the Catch?

- Independently Developed Models Use:
  - ⇒ Different Meshes and Time Scales
  - ⇒ Different Programming Models and Languages
  - ⇒ Different Data Structures and Decompositions

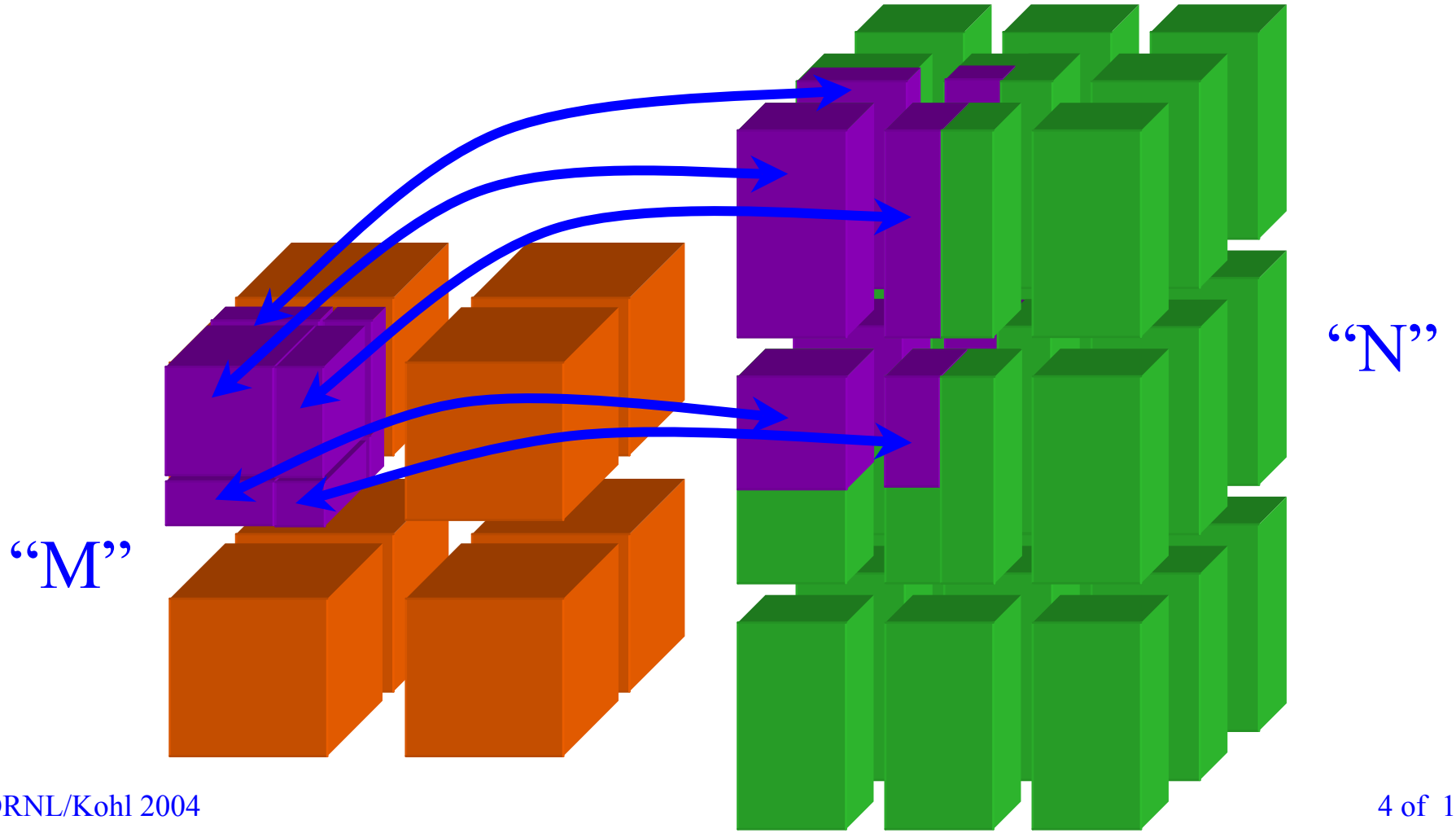


- Coupling Therefore Requires:
  - ⇒ Spatial and Temporal Interpolation
  - ⇒ Language Interoperability & Component Tools
  - ⇒ Parallel Data Exchange and Redistribution
    - Synchronization, Communication Schedules

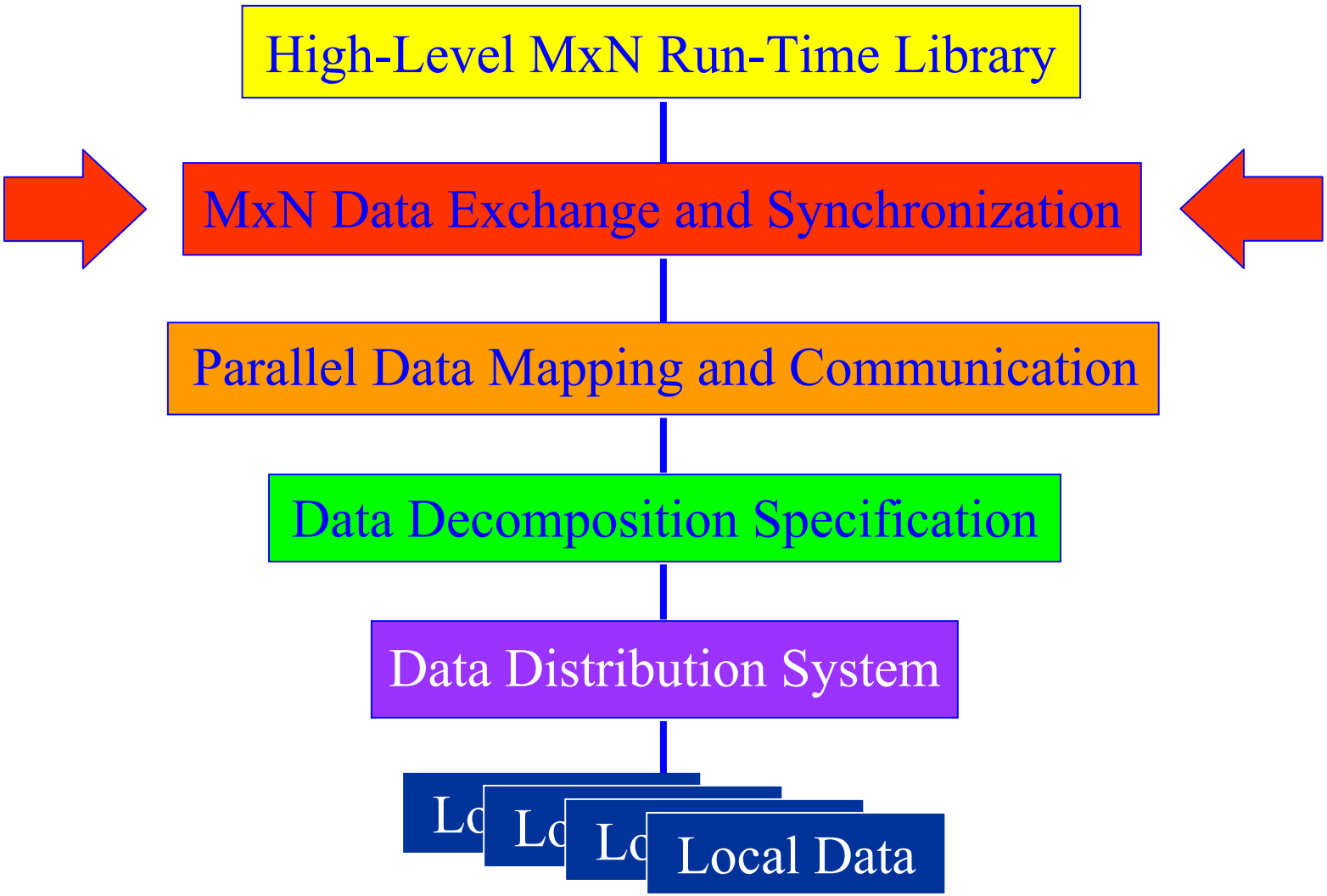




# The "Basic" Problem: MxN Parallel Data Exchange



# MxN Layered Structure




# The Common Component Architecture (CCA)

- Component-Based Software Engineering

  ⇒ Manage Complexity of Scientific Simulation Software

- Successful in Business ~ Corba, DCOM, EJB

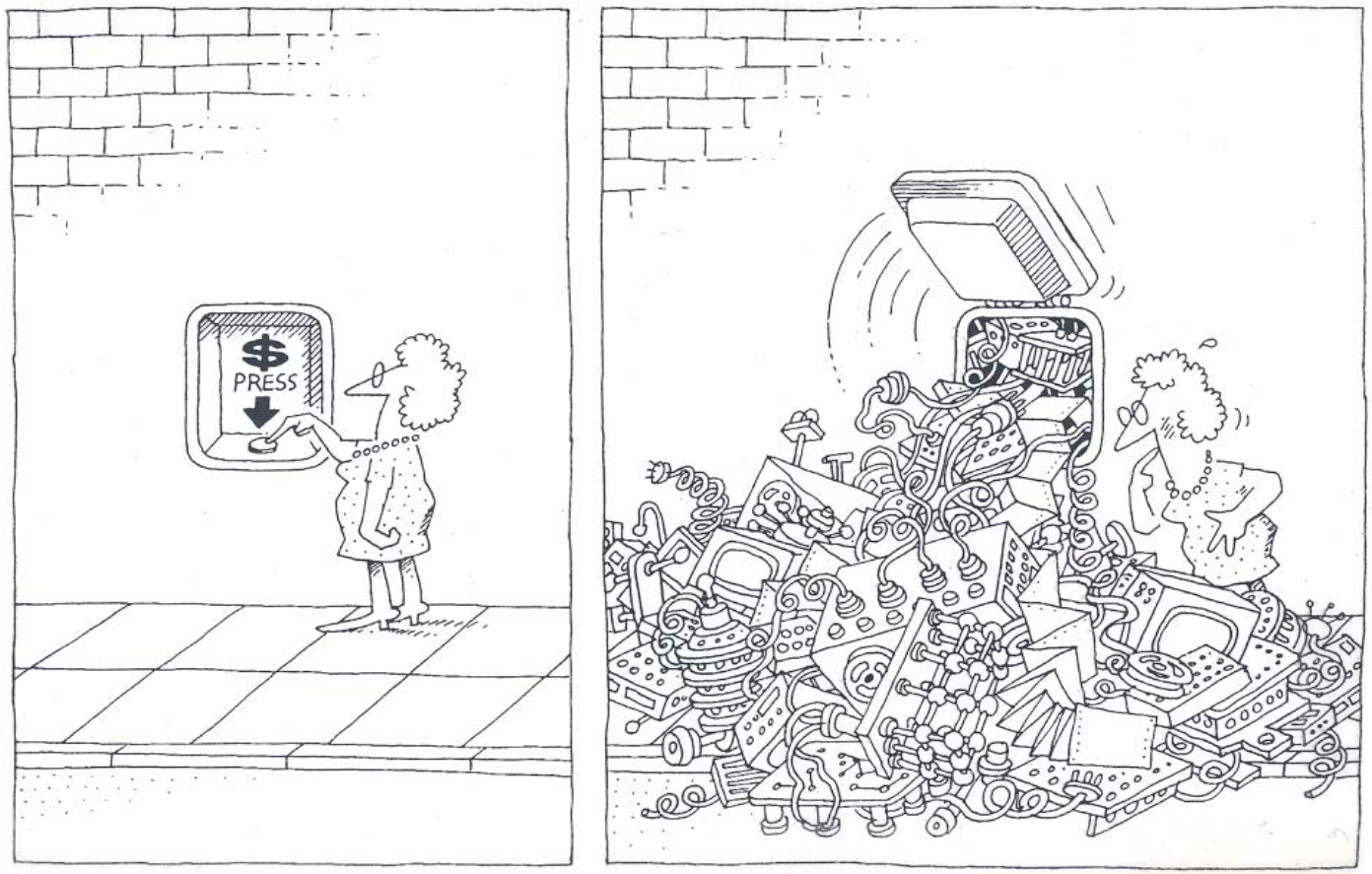
  ⇒ Add Performance, Languages, Science Data Models, and MxN Parallel Data Redistribution!

- National Forum (Open) / DOE SciDAC Center




http://www.cca-forum.org/

# Why Components?



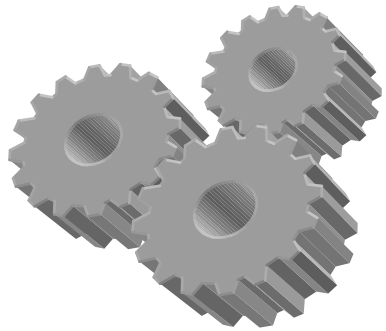

The task of the software development team is to engineer the illusion of simplicity *[Booch]*.

# Why Components?

- Well-Defined Abstract Interfaces
  - ⇒ No "Cheating" Allowed…
    - Hides Implementation Details (Separated Name Spaces)
    - Combine Multiple Implementations in Same Code
  - ⇒ Note: Cost of Standardization Efforts…
- Eases Code Re-use, Enables Swapping
  - ⇒ Simple Wrappers, Bridge Language Gaps
  - ⇒ "Port" Abstraction Enables Intelligent Proxies…

The task of the software development team is to engineer the illusion of simplicity *[Booch]*.

# MxN Interface



## Parallel Data Exchange Operations

"Easy" Stuff

⇒ Describe and "Register" Data/Decompositions

⇒ Map Data Elements ~ "Communication Schedules"

⇒ Build Synchronized MxN "Connections"

⇒ Initiate Data Transfers Asynchronously: "dataReady()"

## Generalizes Existing Tools:

⇒ CUMULVS, PAWS, Meta-Chaos

CUMULVS

PAWS Parallel Application Work Space

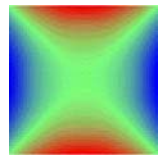
CHAOS

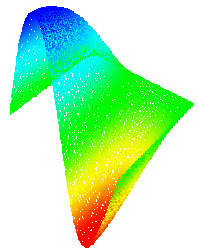
## Several interface evolutions (ongoing…)

⇒ Reconciling appropriate level of detail & flexibility

# MxN "Explicit" Component Solution

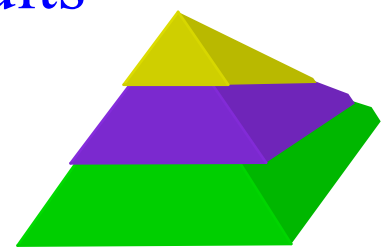
- Port-based direct invocation of MxN methods
  - ⇒ Most general solution, but…
  - ⇒ More challenging to the end-user scientist.
    - "Assembly language" level interface…
    - Preliminary platform for experimentation
- Several implementations have evolved…
  - ⇒ Using generalized DistArrayDescriptor
  - ⇒ Basic inter-framework capabilities
  - ⇒ Visualization, Coupling





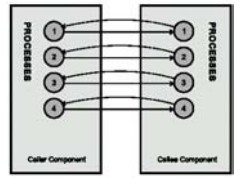
# Future Work ~ "Implicit" Solutions

- Need simpler, high-level interfaces
  - ⇒ For the non-expert… even automated handling
- Targeting built-in framework services
  - ⇒ Capture method invocations via port indirection
  - ⇒ Implicitly apply MxN functions to reconcile parallel data arguments & returned results
- Increases framework complexity
  - ⇒ Use pluggable service registration!
  - ⇒ Requires additional method specifications…

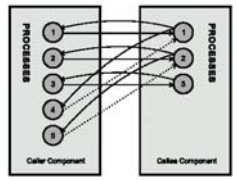


# Parallel Remote Method Invocation (PRMI)

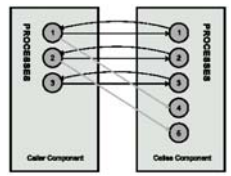
- Next step beyond "simple" data exchange…
  - ⇒ Method itself has parallel context
  - ⇒ Specification of semantics and policies is key!



- Preliminary PRMI progress:
  - ⇒ PAWS prototype and early policy identification
    - Invocation scheduling, marshalling arguments & results
  - ⇒ 2nd SCIRun prototype explores method specification
    - SIDL extensions for "independent" and "collective" methods
    - With sub-grouping, generalizes PRMI invocation semantics

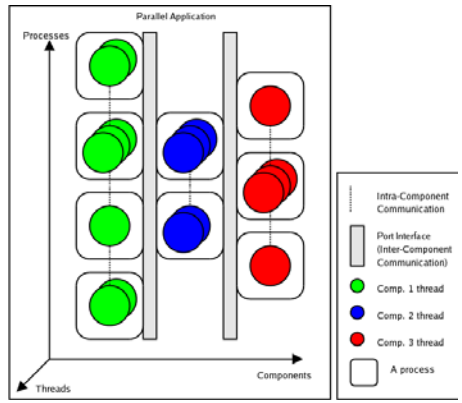


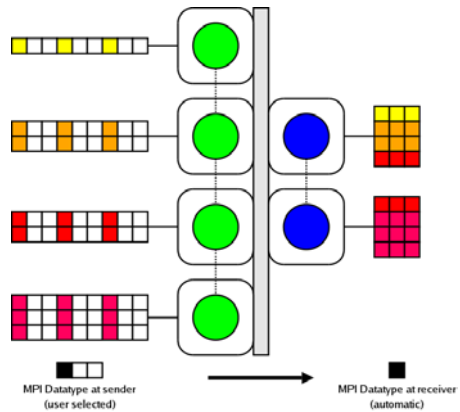
- Still much research ahead…
  - ⇒ Transport mechanisms (SOAP, etc)
  - ⇒ Parallel data argument & results meta-data specification…



# Distributed MxN Data Scenarios

- Incompatible with Parallel MxN
  - ⇒ No co-location components
  - ⇒ Different connection semantics
- Distributed-Parallel Framework
  - ⇒ Experimental ~ "DCA" (Indiana)
  - ⇒ Process Participation & Synch
    - MPI Communicator Groups
  - ⇒ Manual Argument Redistribution
    - Who dictates layout reconciliation?
      - → Caller or Callee…?




Figures © 2004 Felipe Bertrand.

# MxN and Climate (MCT / ESMF)

- Climate-specific coupling models & technology
  - ⇒ Amenable to generalized MxN specification
  - ⇒ Two-way integration underway…
    - Re-package MCT/ESMF as CCA components
    - Build MxN component on top of MCT/ESMF



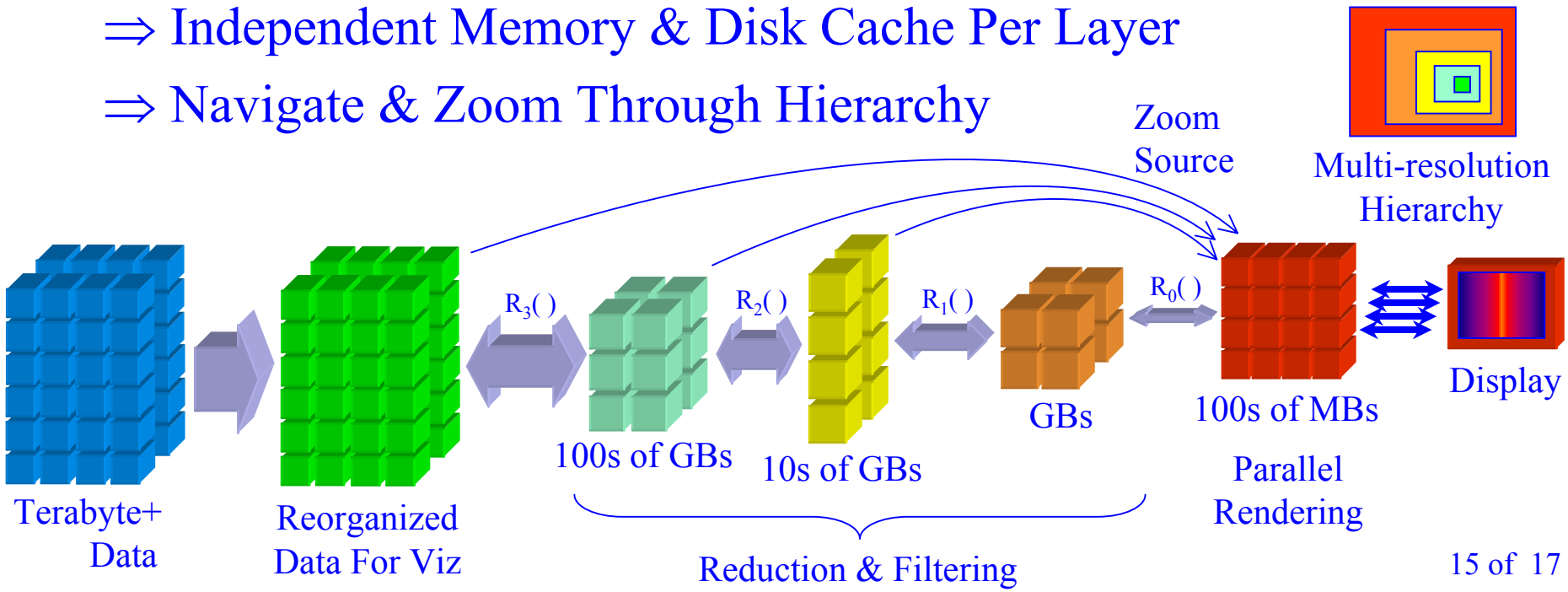
- Ongoing reconciliation of terms and concepts
  - ⇒ Between MxN and MCT, and CCSM and ESMF…
- Componentization of CCSM models
  - ⇒ Atmosphere, ocean, sea-ice, land-surface, river-runoff, plus flux couplers…

# Scalable Visualization Cache Architecture

- Increasingly Massive Scientific Data Sets
  - ⇒ Too Large to Fully Explore / Visualize Interactively
- Modular, Layered Viz Cache Framework
  - ⇒ Parallel Storage, Analysis & Reduction Per Layer
  - ⇒ Independent Memory & Disk Cache Per Layer
  - ⇒ Navigate & Zoom Through Hierarchy

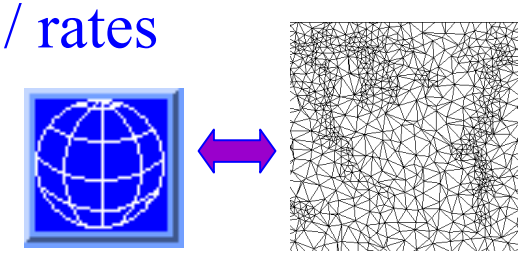
CCA / MxN
CUMULVS

SDM & ASPECT
Data Reduction



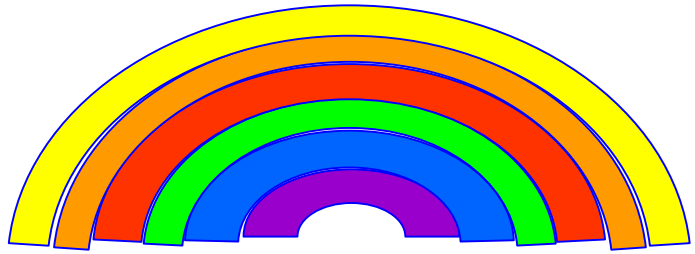
# “Real” Model Coupling…

- MxN parallel data redistribution is just the beginning of real model coupling…
- Need interpolation and data translation
  - ⇒ Spatial ~ different meshes & coordinate spaces
  - ⇒ Temporal ~ different time frames / rates
  - ⇒ Flux Conservation
  - ⇒ Units Conversion



- Must explore composing “filters” with MxN
  - ⇒ Pipeline efficiency and compound “Quality of Service”

# MxN Summary



- Stable MxN specification and component solutions
  - ⇒ Next step ~ implicit framework services
- Parallel Remote Method Invocation (PRMI)
  - ⇒ Initial semantics being defined, much to do…
- Distributed MxN Framework Experiments (DCA)
  - ⇒ Culminating Generalization of Parallel/Distributed…
- Tip of the iceberg for production model coupling
  - ⇒ Need development of suite of interpolation “filters”…

http://www.csm.ornl.gov/cca/mxn/