

2<sup>nd</sup> Biennial, Official, 100% Pure, CCA Approved...

# MxN Working Group Meeting!

(This is your brain...)



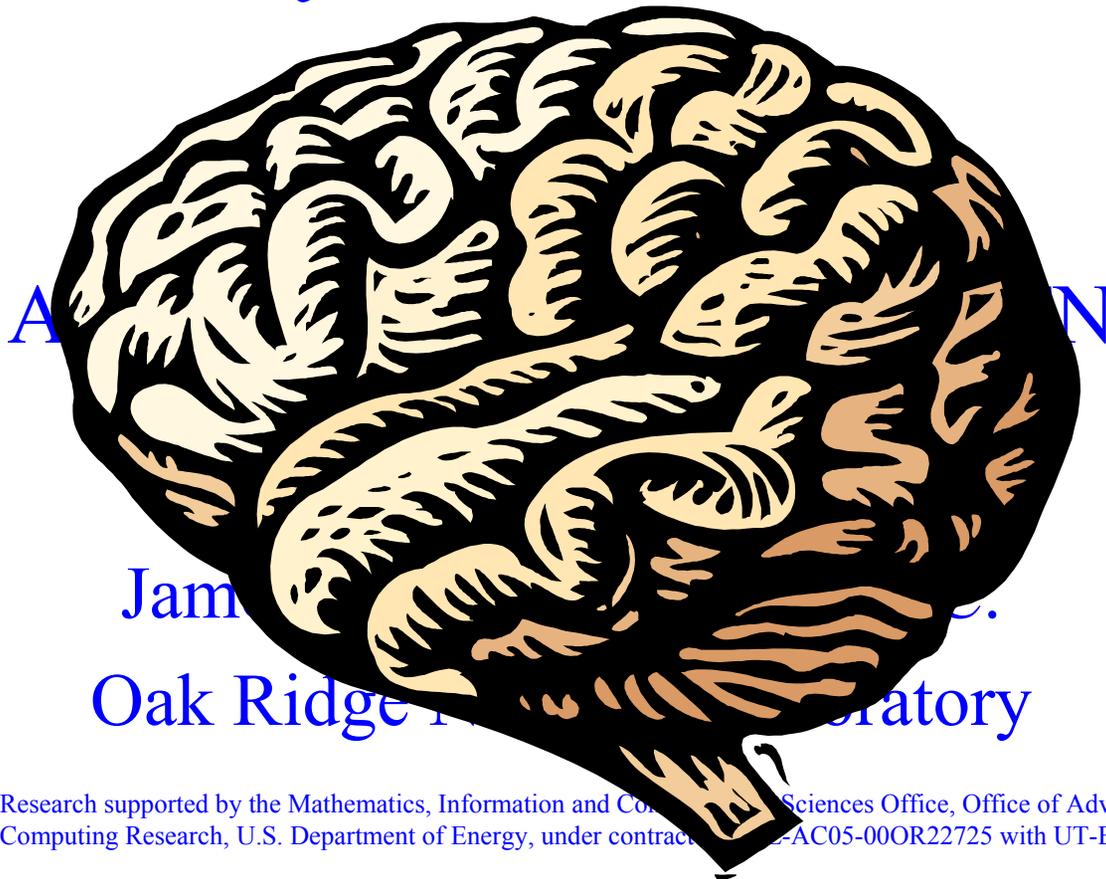
At The Radisson, Knoxville, TN  
January 21, 2004

James “Jeeembo” Kohl, M.C.  
Oak Ridge National Laboratory

2<sup>nd</sup> Biennial, Official, 100% Pure, CCA Approved...

# MxN Working Group Meeting!

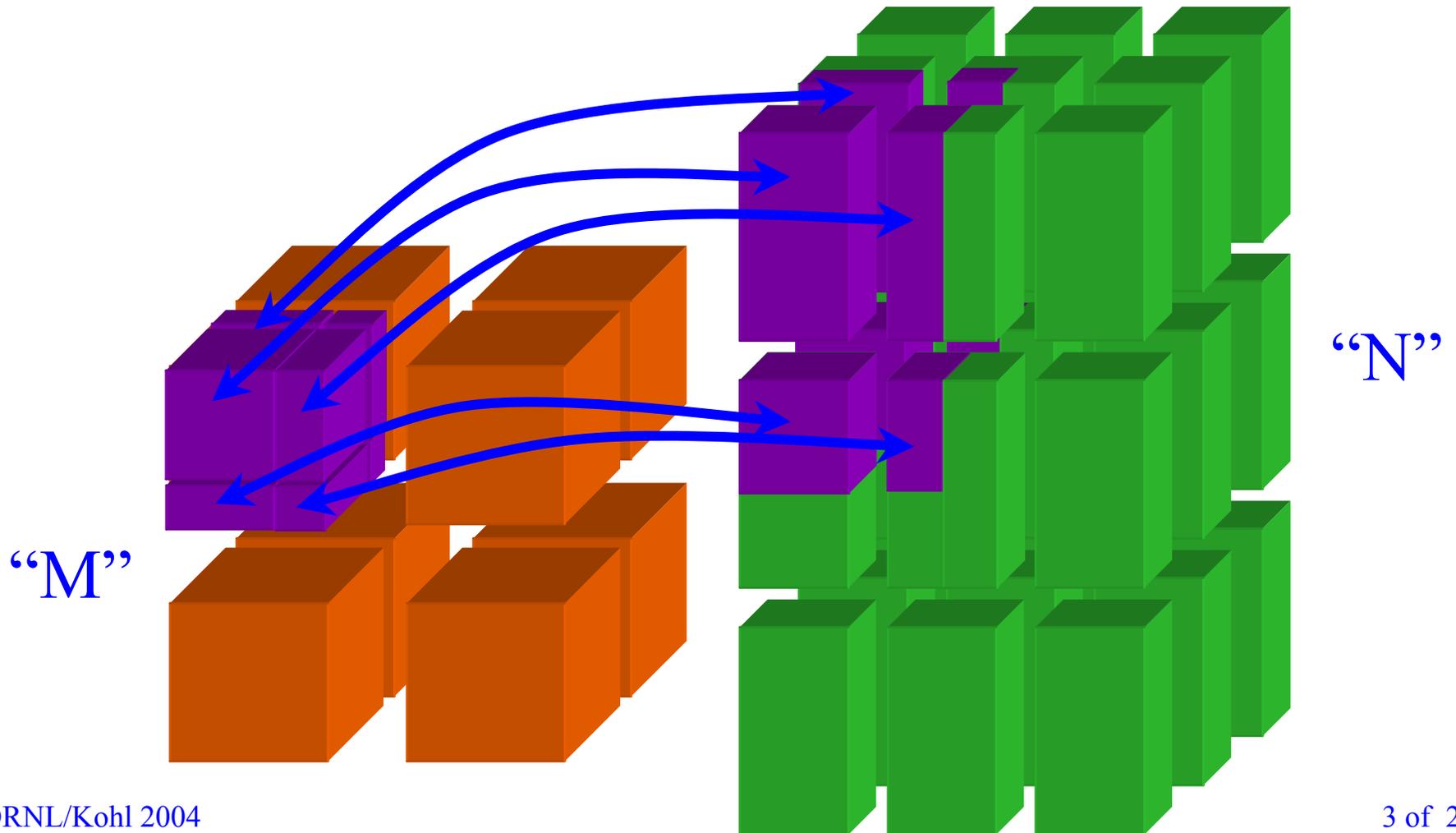
(This is your brain on MxN!)



# MxN Agenda

- CCA MxN Overview (Jeeeeeeeeem)
- ESMF and Data Parallel Issues - Nancy Collins (UCAR)
- DARPA Data Reorganization Effort and the Data Reorganization Interface (DRI) - David Bernholdt (ORNL)
- The UCLA DDB on Mars Time - Dan Katz (JPL)
- Kosta MxN Imitations - Steve Parker (Utah)
- Break
- MPI-Based MxN Framework - Felipe Bertrand (Indiana)
- ECho and MxN - Hasan Abbasi (GA Tech)
- SIDL MxN Specification - Wael Elwasif (ORNL)
- General MxN Discussion...

# The “Basic” Problem: Parallel Data Exchange

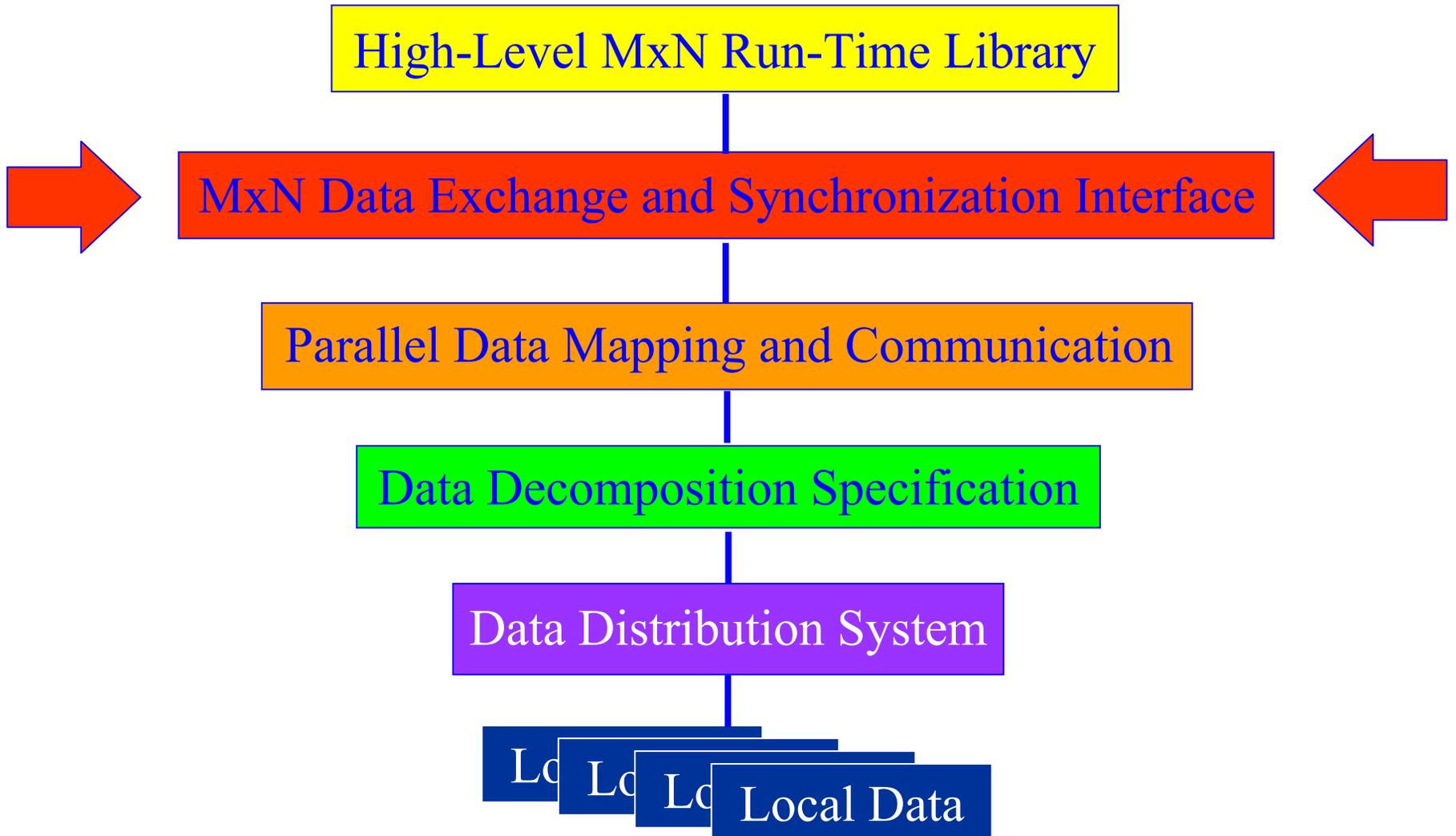


# Cooperating Parallel Components...?!



- CCA-specific capability (non-ad-hoc...)
- Requirement of emerging scientific apps
  - ⇒ Generalized model coupling (multi-parallel)
  - ⇒ Efficient daisy-chaining of numerical libraries
  - ⇒ Visualization (parallel rendering / serial gathering)
- Fundamental issue ~ data decompositions
  - ⇒ Different distribution for each parallel component
  - ⇒ Need to describe/reconcile disparate data organizations
- And, more than just parallel data...
  - ...Parallel method invocations, too!

# MxN Internal Structure

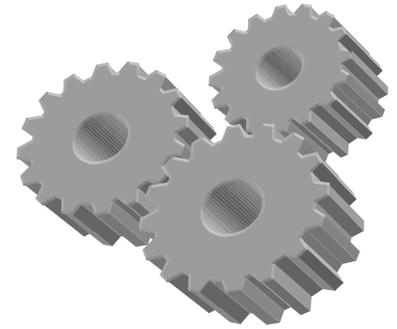


# MxN Progress to Date...



- Generalized MxN specification
  - ⇒ Based on CUMULVS (ORNL) and PAWS (LANL)
- SCMD peer MxN component solution
  - ⇒ Visualization; combustion and climate apps...
- Parallel Remote Method Invocation (PRMI)
  - ⇒ Preliminary semantics and specifications
- Distributed MxN solution
  - ⇒ MPI I/O approach

# Current MxN Specification



## Parallel data exchange operations

- ⇒ Describe and “Register” data / decompositions
- ⇒ Map data elements ~ “Communication Schedules”
- ⇒ Build MxN “Connections”
  - Various synchronization options
- ⇒ Initiate each data transfer, per parallel instance
  - Local dataReady() method suite

## Foundation for generalized model coupling

- ⇒ Need to add spatial and temporal interpolation
- ⇒ Units conversion...

# MxN Interface Origins



- Generalizes many existing tools
  - ⇒ Primarily CUMULVS (ORNL) and PAWS (LANL)
    - Also “DataCutter” at U. Maryland and others...
  - ⇒ Several synchronization models subsumed
    - Point-to-point send & receive (PAWS)
    - Persistent channel w/periodic transfers (CUMULVS)
  - ⇒ Dense, rectangular data distributions supported
    - A la HPF ~ e.g. block, cyclic, collapse
    - Some unstructured mesh experiments (incl. triangular)
- Several interface evolutions (ongoing...)
  - ⇒ Reconciling the appropriate level of detail & flexibility
  - ⇒ Now Split Several Ways ~ Data, Control, Handles...



# MxN Data “Registration”

- Tell the “MxN” handler about the data:
  - ⇒ Local Information ~ Ptr, Type, Size
  - ⇒ Distributed Data Decomposition

**Each A:**

```
“Data” Adata;
```

```
registerData( in Adata, True, ReadOnly,  
             out Adhandle );           (Synch)
```

```
unregisterData( Adhandle );
```

# MxN Data “Handle”

- Object for wrangling “MxN Data” info:

```
set/getAppName( string applname );
```

```
set/getDataName( string name );
```

```
set/getClientData( void *ptr );
```

```
set/getDataBounds( int *upper, int *lower, int rank );
```

```
set/getDADF( dadf.DistArrayDescriptor dscr );
```

# MxN Data “Synch”

- Means for basic initialization of MxN system:
  - ⇒ All initial data fields have been registered...
    - Simple barrier to ensure everyone is “ready”...

```
synch( );
```

# Data Ready

- Each Parallel Entity Activates Local Transfer
  - ⇒ “dataReady( )” Releases Local Data for Use
  - ⇒ Indicates Parallel “Consistency” of Data
  - ⇒ Data Elements Transferred As Available...

All A (not necess synch):

```
dataReady( in Adhandle, True, ReadWrite );
```

All B (not necess synch):

```
dataReadyBegin( in Bdhandle, True, ReadWrite );
```

```
. . .
```

```
dataReadyTest( in Bdhandle );
```

```
. . .
```

```
dataReadyEnd( in Bdhandle );
```

# MxN Communication Schedules

- Map Together Two Parallel Entities
  - ⇒ Fundamentally, Element by Element...
  - ⇒ Based on Data Decomposition Info

All A:

```
createCommSched( in dstData, in srcData, out csHandle );
```

```
freeCommSched( in csHandle );
```

```
extrapolateCommSched( in csHandle, in dstData, in srcData,  
                      out new_csHandle );
```

# MxN Connections

- Connect Up Two Parallel Entities  
⇒ Commence Synchronization

All A:

```
makeConnection( dstDhandle, srcDhandle, csHandle,  
               NoSynch, Default, 4, 5, cHandle );  
               (Src Synch)  (Dst Synch) (Src & Dst (Out)  
                               Freqs)
```

```
releaseConn( cHandle );
```

```
changeFreqs( cHandle, dstFreq, srcFreq );
```

# MxN Parallel Data Transfers

- Request Actual Data Transfer  
⇒ Either Side, or 3<sup>rd</sup> Party Component Control

All A:

```
requestTransfer( in cHandle, out tHandle );
```

```
do {
```

```
    . . .
```

```
    testTransfer( tHandle );
```

or

```
    waitTransfer( tHandle );
```

```
    . . .
```

```
} while ( );
```

# Hooking It All Together!

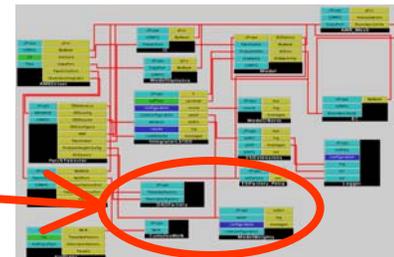
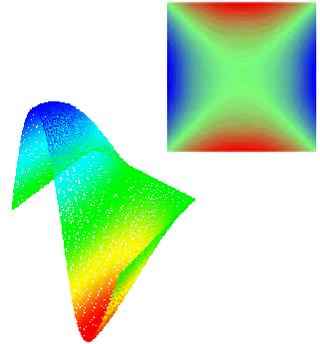
- Discovery of other data to couple with...
  - ⇒ Obtain handles for previous control methods

```
getRegisteredData( in string dataName, out dHandle );
```

```
getAllRegisteredData(  
    out array<DataHandle, 1> dHandles );
```

# MxN “Explicit” Component Solution

- Port-based direct invocation of MxN methods
  - ⇒ Most general solution, but...
  - ⇒ Most challenging to the end-user scientist.
    - “Assembly language” level interface...
    - Preliminary platform for experimentation, higher-level functs
- Several demonstrations (SC00, SC01, SC02)
  - ⇒ Increasingly elaborate visualization solutions
    - Incorporated generalized data description since SC01 (DADF)
  - ⇒ Two main prototypes (reused several apps):
    - CumulvsMxN ~ first inter-framework component capabilities!
    - PawsMxN ~ ping-pong coupling experiments



# MxN and Combustion (CFRFS)

- No coupling needs, yet useful for adaptive chemistry and post-processing...

⇒ Replace DADF with “ParticleCollection”

- Map adaptive mesh patches to “Particle” container

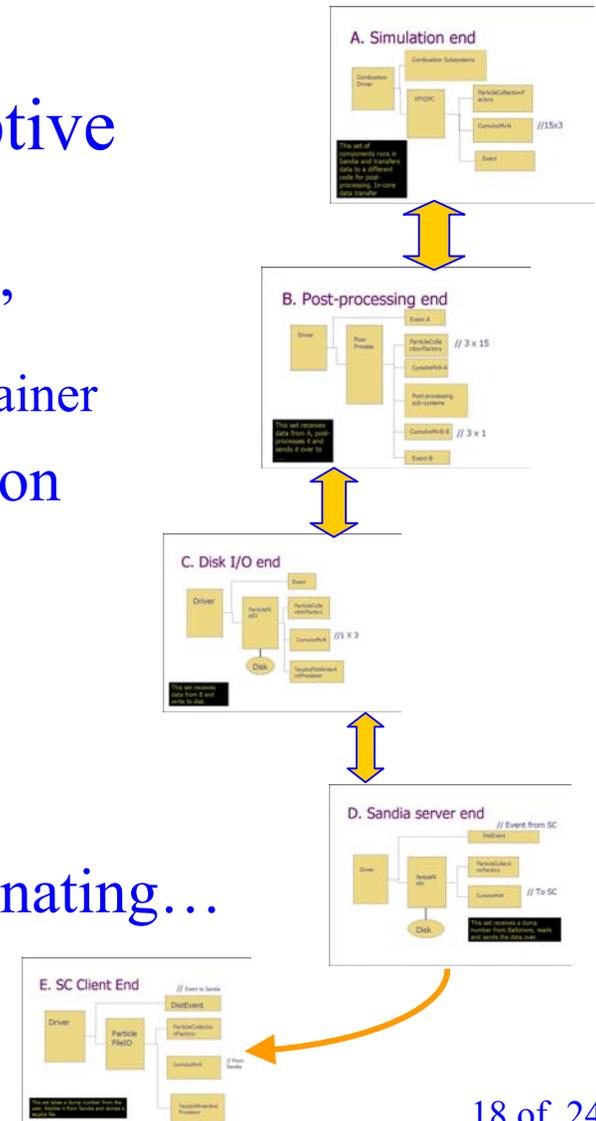
⇒ Interactively offload data for simplification

- “Real” parallel-to-parallel MxN usage... ☺

⇒ Connect several computational clusters

- Plus front-end visualization node(s)
- Glues together multiple SCMD frameworks

⇒ Initial “CumulvsMxNp” prototype culminating...



# 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

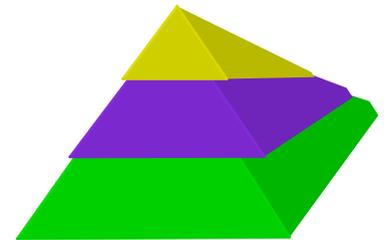
⇒ Between MxN and MCT, and CCSM and ESMF...

- Componentization of CCSM models

⇒ Atmosphere, ocean, sea-ice, land-surface, river-runoff,  
plus flux couplers...

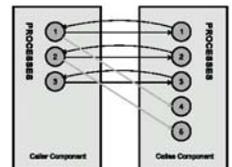
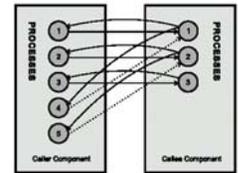
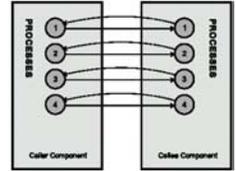
# MxN Future Work ~ Implicit Solutions

- Need simpler, high-level interfaces
  - ⇒ For the non-expert... even automated handling
- Target built-in framework services
  - ⇒ Capture method invocations via port indirection
  - ⇒ Implicitly apply MxN functions to reconcile parallel data in method arguments & 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
  - ⇒ 2<sup>nd</sup> 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 and results specifications...

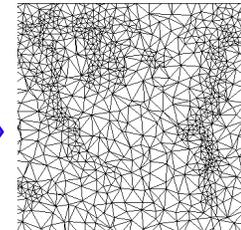


# Distributed MxN Data Scenarios

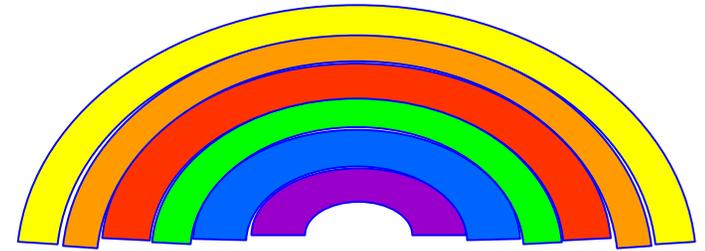
- Large body of distributed HPC applications
  - ⇒ E.g. sensor networks...
- SCMD MxN solutions incompatible...
  - ⇒ No co-location of components in distributed world
  - ⇒ Fundamental differences in connection semantics
- Early exploration (Indiana) ~ MxN via MPI-I/O
  - ⇒ Stream-based communication paradigm
    - Analog to file-based exchanges in serial codes
    - Parallelization of data transfers hidden internally
  - ⇒ Eases integration of existing applications
    - Also useful for unit testing of application components

# “Real” Model Coupling...

- MxN parallel data redistribution is just the beginning of real model coupling...
- Need interpolation and data conversion
  - ⇒ Spatial ~ different meshes & coordinate spaces
  - ⇒ Temporal ~ different time frames / rates
  - ⇒ Flux Conservation
  - ⇒ Units conversion
- Must explore composing “filters” with MxN
  - ⇒ Pipeline efficiency and “super-components”...



# MxN Summary



- Stable specification and component solutions
  - ⇒ Next step ~ implicit framework services
- Parallel Remote Method Invocation (PRMI)
  - ⇒ Initial semantics being defined, much to do...
- Distributed MxN experiments
  - ⇒ Prototypes using MPI-I/O
- Tip of the iceberg for production model coupling
  - ⇒ Need development of suite of interpolation “filters”...

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