

The Latest & Greatest on MxN

(Or How I Decomposed My Speedo... :-o)

Jeeembo Kohl

Oak Ridge National Laboratory

August 24, 2000

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.

“The Plan”

- Overview of the Latest CCA / MxN Appls
- Proposals for an Actual Collective Spec
 - ⇒ Enough Playing Around, Time to Just Do It... 😊
- MxN Conference Paper?
 - ⇒ Tell the World How Brilliant We Are(n't)

CCA / MxN Applications at ORNL

- GIST ~ Phil LoCascio, LSD
 - ⇒ a la CCA Proposal, Fault Tolerance, etc.
- NWCHEM ~ David Bernholdt, CSMD
 - ⇒ Improve Component Flexibility / Coupling
- Virtual Watershed ~ CSMD, ESD, CPED
 - ⇒ Develop Generic “Flux Coupler”
 - ⇒ Leads Into Virtual Human, Other Applications

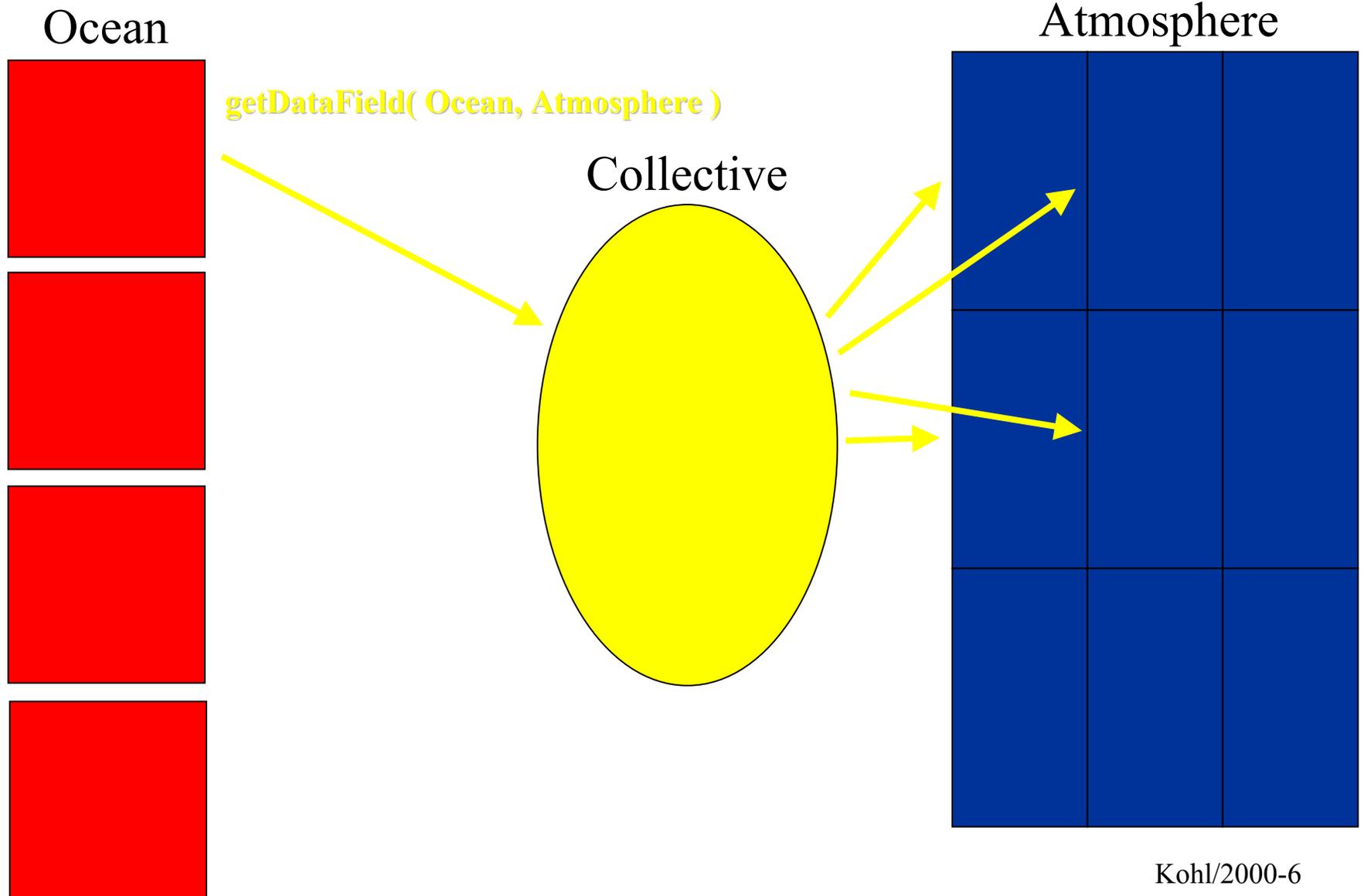
MxN Specification Proposal

- Attempt to Solidify Ideas ~ Concrete Specs
- Parallel Data Exchange & Translation
 - ⇒ Explicit Methods (Implicit Just Needs Semantics)
- Preliminary Data / Decomposition
 - ⇒ Just Rectilinear Meshes
- Initial Interpolation / Conversion Spec Ideas...

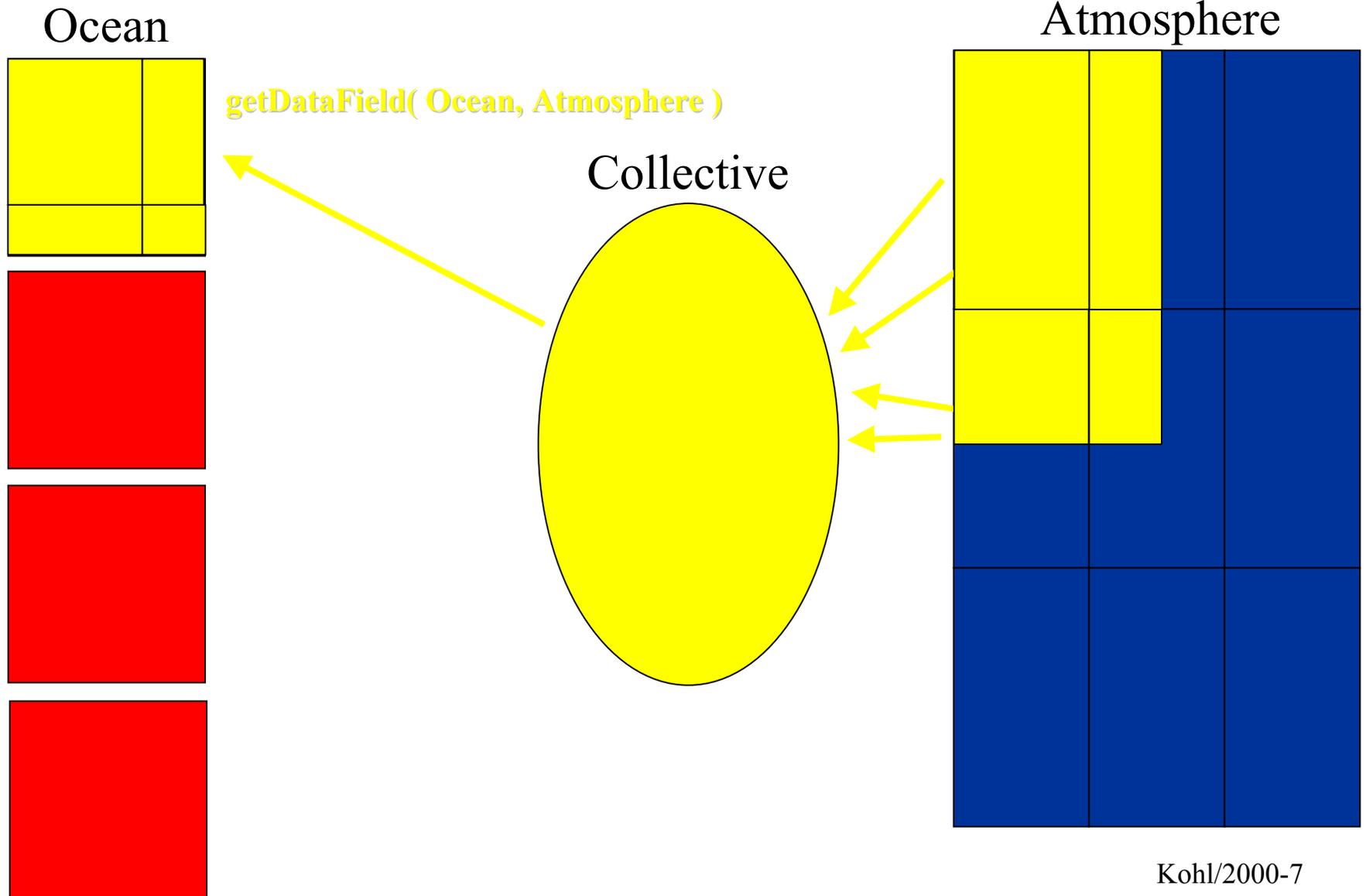
Main Collective Specification

```
package gov.CCA.Collective {  
    interface CollectivePort extends gov.CCA.Port {  
        void addDataField( in DataField field ); // hook up this data field  
        void removeDataField( in DataField field ); // un-hook data field  
        void getDataField( in DataField dst_field, // our data field  
                           in DataField src_field ); // the other data field  
        // DataField encodes Space & Time Information...  
    }  
    ...  
}
```

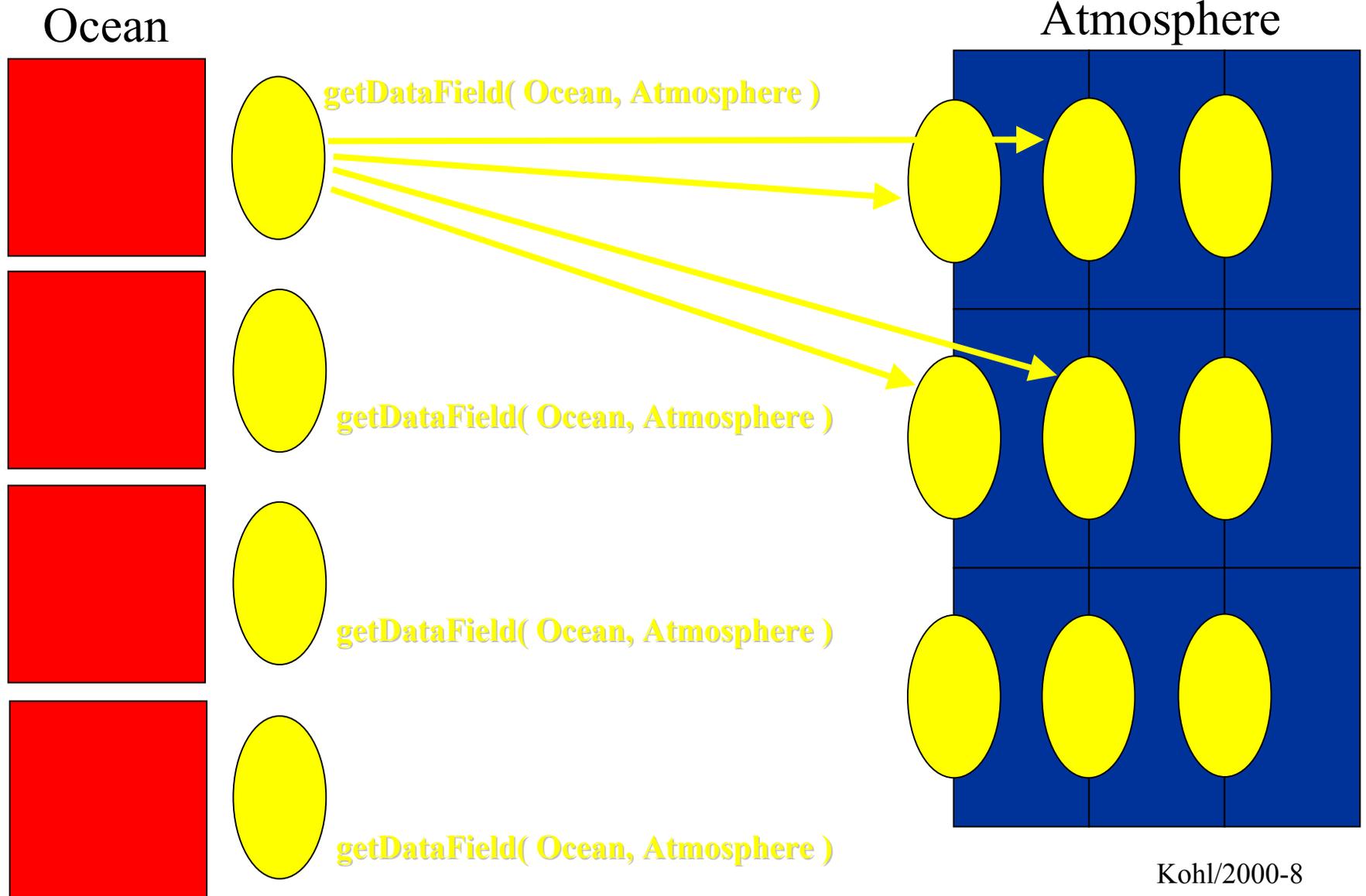
Collective Component Example



Collective Component Example



Collective Component Example



Data Field Specification

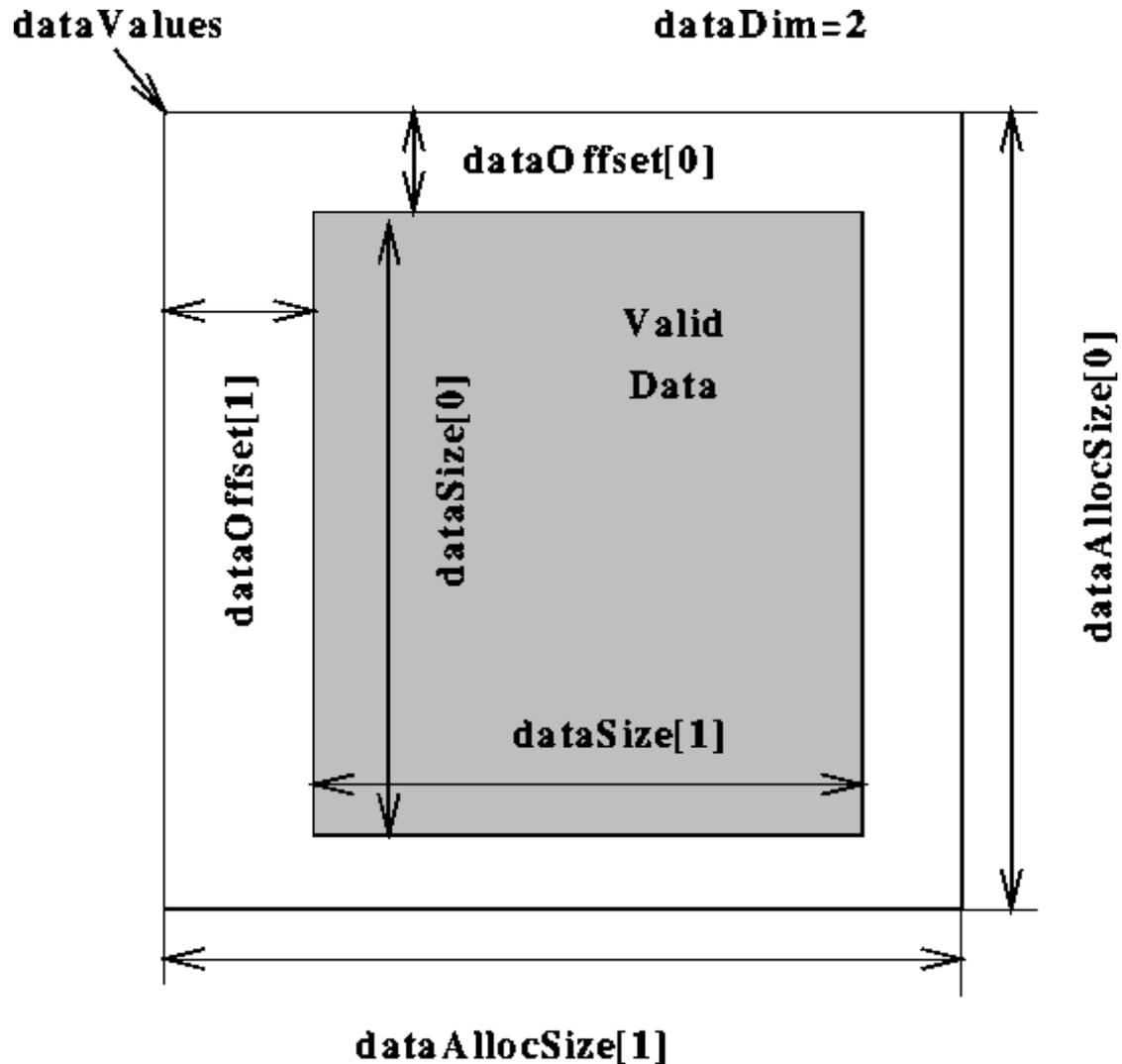
```
package gov.CCA.Collective {      ...
    interface DataField {
        void setField( in string name, // logical name of array
                       in Data data ); // reference to general data spec...
        void setProcArray( ProcArray parray ); // resources holding data field
        void setUnits( in string unitScale, // scale of units, e.g. “meters”, “kg”...
                      in string unitPrefix ); // unit prefix, e.g. “pico”, “deca”, “tera”...
        void setGrid( in string gridType ); // e.g. “rectangular”, “spectral”...
        void setTime( in Time time ); // set current simulation time for data field
        void setPeriod( in Time period ); // simulated time per iteration
        void setReady( // data field is ready for collection
                      in boolean incrTime ); // increment field time (by period) or not
    }
    ...
} ORNL
```

Rectilinear Mesh Data Spec

```
package gov.CCA.Collective {    ...
    interface RectData {
        void setData( in void *dataptr ); // pointer to actual data storage
        void setType( in string dataType ); // basic data type of data
        void setSize( in integer dataDim, // dimension of data array
            in array<integer,dataDim> dataSize ); // cardinality of each axis
        void setOffset( in integer dataDim,
            in array<integer,dataDim> dataOffset ); // valid data each axis
        void setValidData( in integer dataDim,
            in array<integer,dataDim> dataValidSize ); // valid size each axis
        void setDecomp( RectDecomp decomp ); // data distribution for data field
    }
    ...
}
```

ORNL

Local Allocation



Rectilinear Decompositions

```
package gov.CCA.Collective {      ...
    interface RectDecomp {
        void setDecompBounds(
            in integer dataDimGlobal, // dim of global array
            in array<integer,dataDimGlobal> globalLowerBounds,
            in array<integer,dataDimGlobal> globalUpperBounds );
        void setDecompAxis(
            in integer axisIndex, // index of axis x=0, y=1, z=2, etc...
            in RectDecompInfo info ); // info for decompType...
    }
    ...
}
```

Rectilinear Decomp Axis Info

```
package gov.CCA.Collective {    ...
    interface RectDecompInfo {
        void setDecompType( in string decompType );
            // e.g. "Block", "Cyclic"...
        void setDecompProperty( in string name, in string value );
            // e.g. "Block Size", "10"
        void addBounds( in integer numBounds, // # of explicit region bounds
            in array<integer,numBounds> lowerBounds,
            in array<integer,numBounds> upperBounds );
    }
    ...
}
```

ProcArray

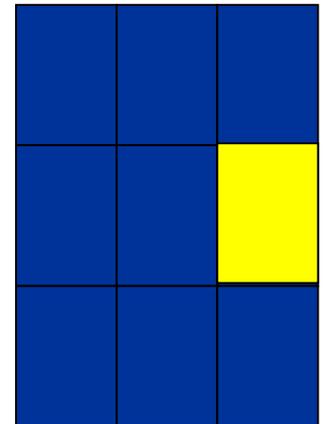
```
package gov.CCA.Collective {    ...
    interface ProcArray {
        void setProcShape( in integer procDim, // dim of processor array
                           in array<integer,procDim> procShape ); // size of each axis
        void setProcAddress( in integer procDim,
                             in array<integer,procDim> procAddress );
                               // address of instance in array
    }
    ...
}
```

Time Map

```
package gov.CCA.Collective {    ...
    interface Time {
        void setTime( in float timeValue, // set time value
                      in string timeUnitScale, // e.g. “seconds”, “minutes”...
                      in string timeUnitPrefix ); // e.g. “pico”, “giga”...
        void incrTime( in float timeValue, // increment time value
                      in string timeUnitScale,
                      in string timeUnitPrefix );
    }
    ...
}
```

Example Data Field Specification

```
FOO.RectData.setData( my_data_pointer );  
FOO.RectData.setSize( 2, { 10, 10 } );  
FOO.Decomp.setDecompBounds( 2, { 30, 30 } );  
FOO.DecompInfo.setDecompType( "Block" );  
FOO.DecompInfo.setProperty( "Block Size", 10 );  
FOO.Decomp.setDecompAxis( 0, "Block", FOO.DecompInfo );  
FOO.Decomp.setDecompAxis( 1, "Block", FOO.DecompInfo );  
FOO.RectData.setDecomp( FOO.Decomp );  
FOO.DataField.setField( "Heat Flux", FOO.RectData );  
FOO.ProcArray.setProcShape( 2, { 3, 3 } );  
FOO.ProcArray.setProcAddress( 2, { 1, 2 } );  
FOO.DataField.setProcArray( FOO.ProcArray );
```



Interpolation Hooks

```
package gov.CCA.Collective {    ...
    interface Interpolate {
        void setInterpTemporal( in function interpolateTime(
            in TimeQueue scratch, in DataField dstField, in DataField srcField ) );
        void setInterpSpatial(
            in string dstGridType, in string srcGridType,
            in function interpolateSpace(
                in DataField dstField, in DataField srcField ) );
    }
    interface TimeQueue {    // circular buffer...
        void setTimeQueueLength( in integer queue_size );
        void addDataFieldToQueue( in DataField srcField );
    }
}
```

ORNL

Units Conversion Hooks

```
package gov.CCA.Collective {    ...
    interface Convert {
        void setUnitConversion( in string dstUnits, in string srcUnits,
                                in function convertUnits(
                                    in void *dst_dataptr, in void *src_dataptr ) );
    }
}
```

MxN Conference Paper?

- Seems to be Plenty to Present Now... 😊
- Which Conference?
- Topics:
 - ⇒ Data Decomposition Specification
 - ⇒ Parallel Data Exchange Issues
 - ⇒ Collective Specification(s)...
 - ⇒ Future Issues ~ Interpolation, Unstructured
- Authors:
 - ⇒ Jeembo, Scott, Kate
 - ⇒ Data Definition Folks (Ben, Paul, Lori)?

JEEMBO'S DONE

You Can Wake Up Now... ☺