



Managing Scientific Data *Efficiently* in a DBMS

Introduced by:

Christine D'Angela (IBM Corporation)

Presented by:

Ian Barrodale (Barrodale Computing Services - BCS)

and

Robert Uleman (IBM Corporation)

Date:02/18/2009 Time:09:00 - 10:00 PST (GMT-0800)

BCS



Managing Scientific Data *Efficiently* in a DBMS

In this web conference, you'll learn about:

- 1. Housing scientific data in a DBMS vs. flat files**
- 2. A new tool for putting scientific data into a DBMS**
- 3. DataBlades – DBMS extensions to enhance performance**
- 4. Grid DataBlade – flexible & efficient DBMS management of multi-dimensional gridded datasets**
- 5. DBXten DataBlade – faster DBMS reads & indexing using less disk space**

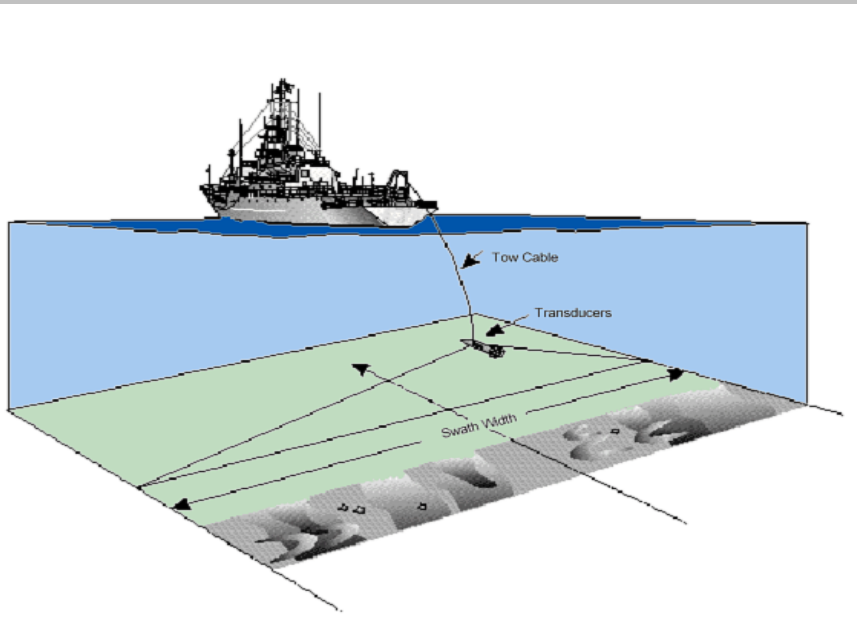
Barrodale Computing Services Ltd. (BCS)

“At BCS we let the actual tasks that our customers are trying to accomplish guide our solutions, rather than producing software that dictates how customers can perform their work.”

BCS develops software and provides R&D services to scientists and other technical customers

BCS has completed 450+ software development jobs over the past 31 years

BCS is an IBM Advanced Business Partner and has a long-term professional staff with wide experience





Flat Files or a DBMS?

Some scientists and technologists prefer to keep their data in flat files and avoid using a database management system (DBMS). Our purpose in this webinar is to show such users how a DBMS can provide productivity benefits.

Fastest time is **green**, next fastest is **blue**, and the slowest time is **red**

| Nature of Extraction | Extraction Size | DBXten DataBlade | Grid DataBlade | NetCDF flat files |
|-----------------------|-----------------|------------------|----------------|-------------------|
| Time profile | 81x1x1x1 | 1.03 sec | 2.24 sec | 1.30 sec |
| Depth profile | 1x66x1x1 | 0.07 sec | 0.17 sec | 0.72 sec |
| Planar slices | 1x1x50x50 | 0.06 sec | 0.09 sec | 0.03 sec |
| | 1x1x100x100 | 0.09 sec | 0.13 sec | 0.03 sec |
| | 1x1x200x200 | 0.17 sec | 0.23 sec | 0.04 sec |
| Depth-Latitude slices | 1x33x25x1 | 0.07 sec | 0.16 sec | 0.37 sec |
| | 1x33x50x1 | 0.09 sec | 0.20 sec | 0.38 sec |
| | 1x66x100x1 | 0.14 sec | 0.38 sec | 0.80 sec |
| | 1x66x200x1 | 0.20 sec | 0.59 sec | 0.68 sec |
| 4D Volumes | 10x5x20x40 | 0.19 sec | 0.46 sec | 0.69 sec |
| | 10x10x40x80 | 0.34 sec | 0.99 sec | 1.24 sec |
| | 10x20x80x160 | 1.12 sec | 3.75 sec | 2.49 sec |
| Average time | | 0.30 sec | 0.78 sec | 0.73 sec |



Why Some Scientists Don't Use Databases

The main DBMS criticisms from a customer survey that BCS conducted in 2008 were:

- 1. “It’s not easy to get my data into a database.”**
- 2. “Databases don’t have all the features that I need.”**
- 3. “Databases are too slow and too big.”**

Let’s address these issues in order.



Loading Data into Databases

BCS has now produced DaL, a *free* Java database client for loading the contents of scientific data files into standard relational databases.



Draw and Load (DaL)

- **By drawing lines between graphic nodes representing file contents and database tables, a user controls how data is stored**
- **The current version (Feb2009) of DaL works with Informix and PostgreSQL**
- **DaL has support for both BCS database extensions (DBXten and Grid DataBlades), but it can also be used without them**
- **DaL uses Unidata's NetCDF Java library and so it can read from any Common Data Model file**

Common Data Model

- **CDM supported file formats:**
 - **NetCDF-3, NetCDF-4, NetCDF-5**
 - **GINI (GOES Ingest and NOAAPORT Interface) image format**
 - **GEMPAK gridded data (as of version 2.23)**
 - **DMSP (Defense Meteorological Satellite Program)**
 - **NEXRAD Radar level 2 and level 3**
 - **ATD's DORADE radar file format**
 - **HDF4, HDF5**
 - **GTOPO 30-sec elevation dataset (USGS)**
 - **CINRAD level II (Chinese Radar format)**
 - **McIDAS AREA**
 - **Universal Radar Format**
 - **USPLN and NLDN lightning data**
 - ***GRIB version 1 and version 2**
 - ***HDF-EOS and HDF5-EOS**
 - ***BUFR**
- (* ongoing work)



Common Data Model

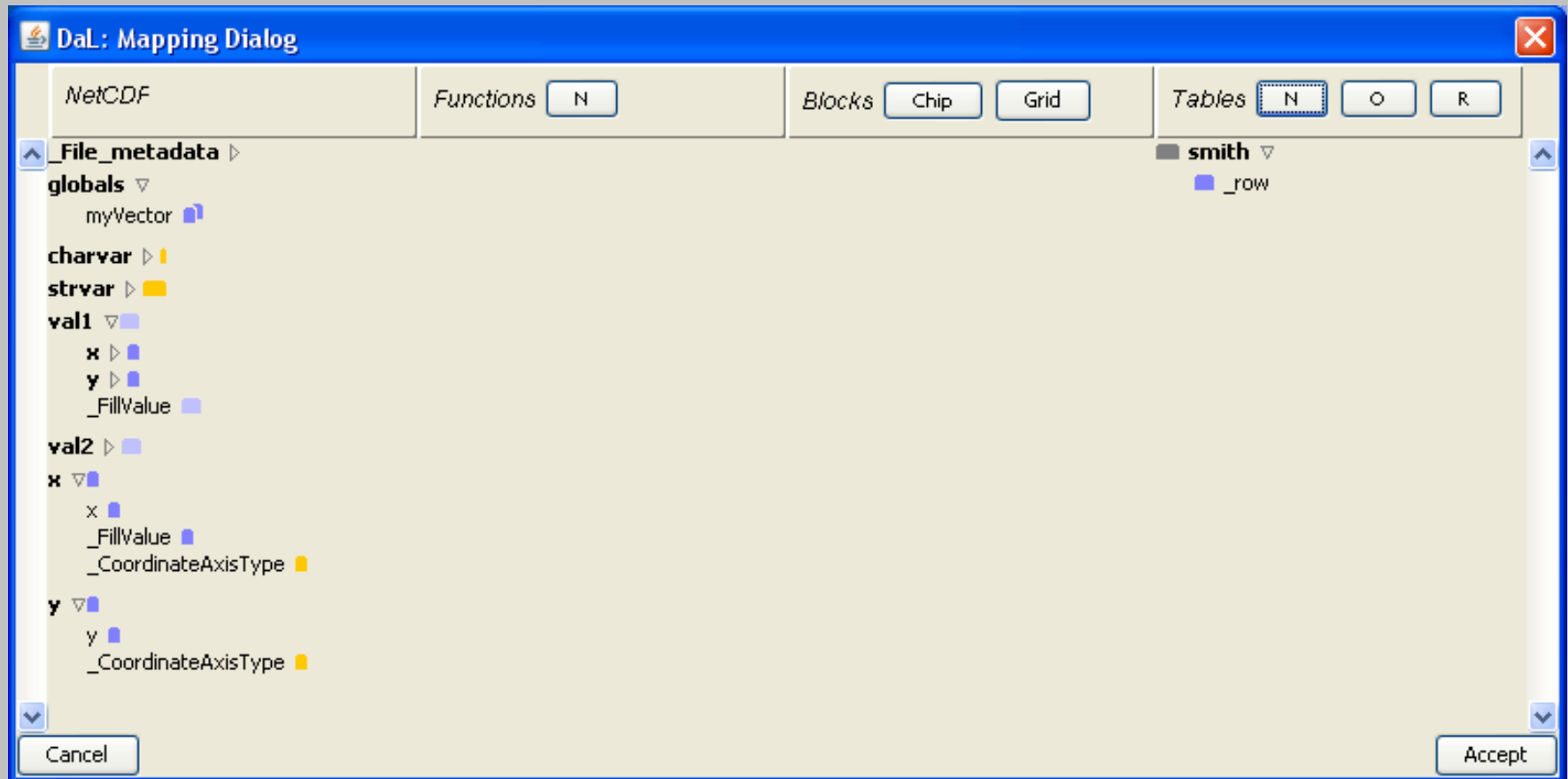
- **CDM supported client/server protocols:**
 - **OPeNDAP client/server protocol**
 - ***ADDE client/server protocol**
(* limited support)



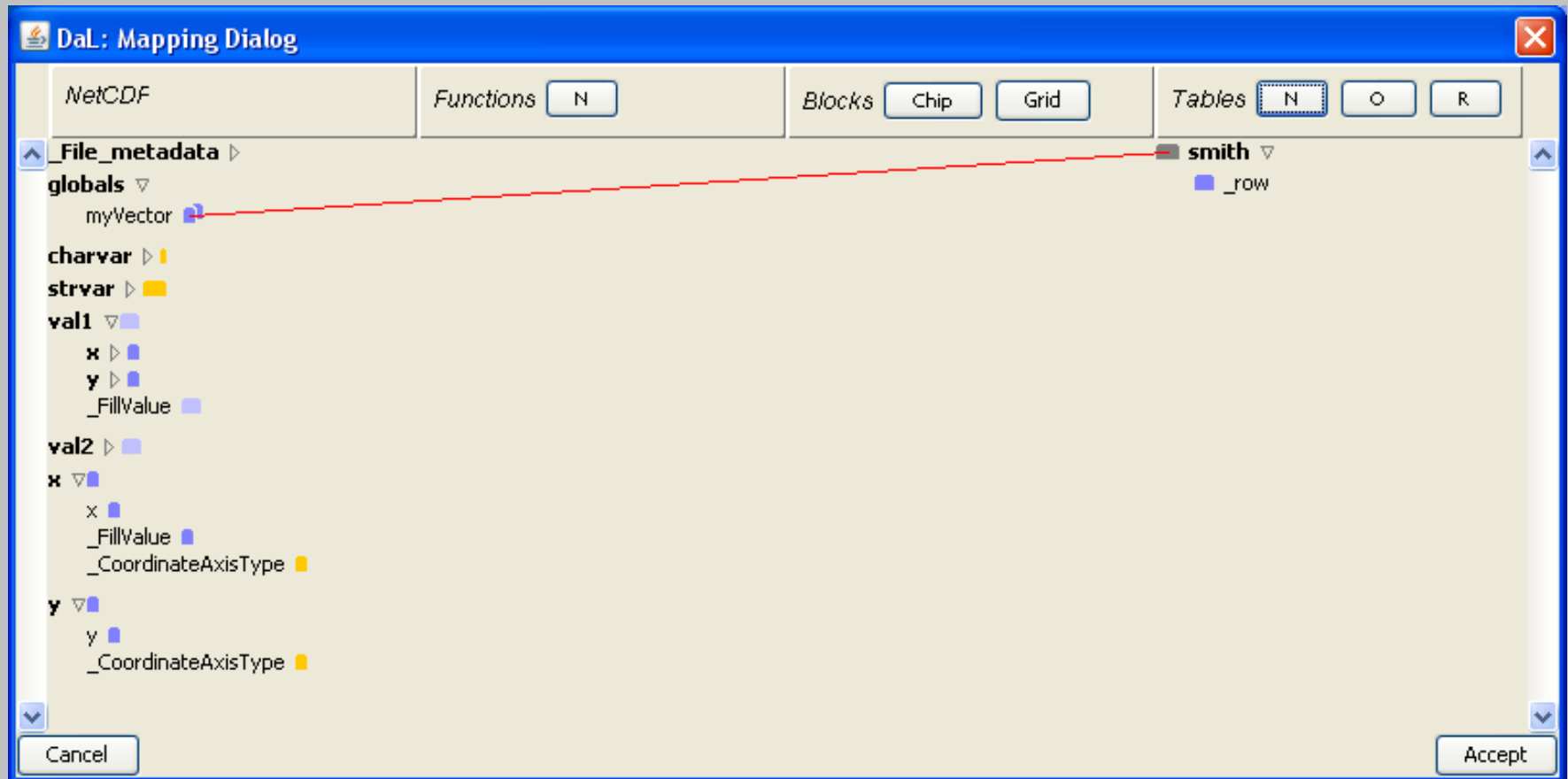
Draw and Load (DaL) Demo

Using a simple example, the next five slides demonstrate how DaL can be used to load scientific data into a database by just drawing lines between graphic nodes that represent file contents and database tables.

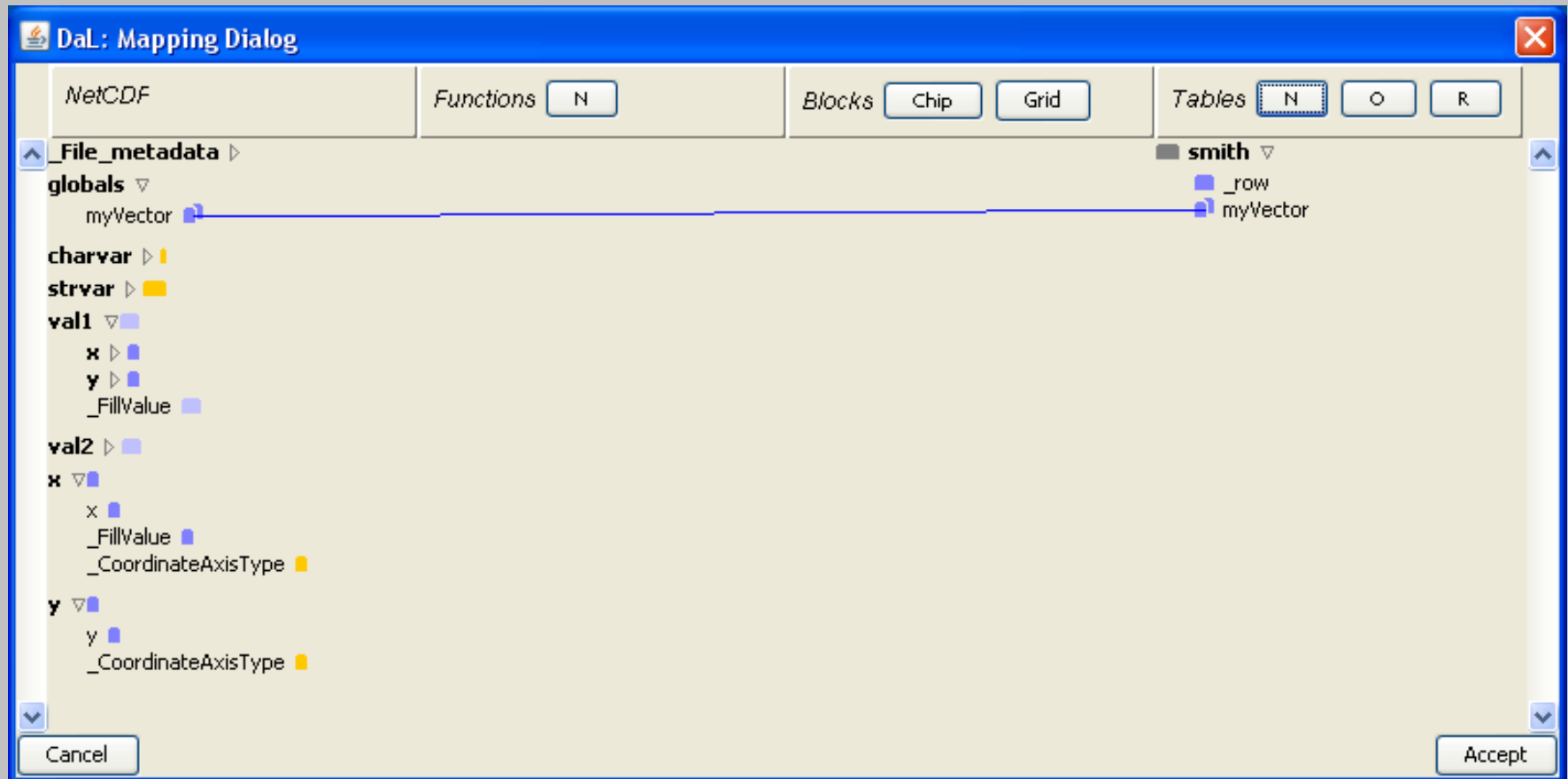
DaL - before any connections are drawn



DaL - connecting a variable to a table



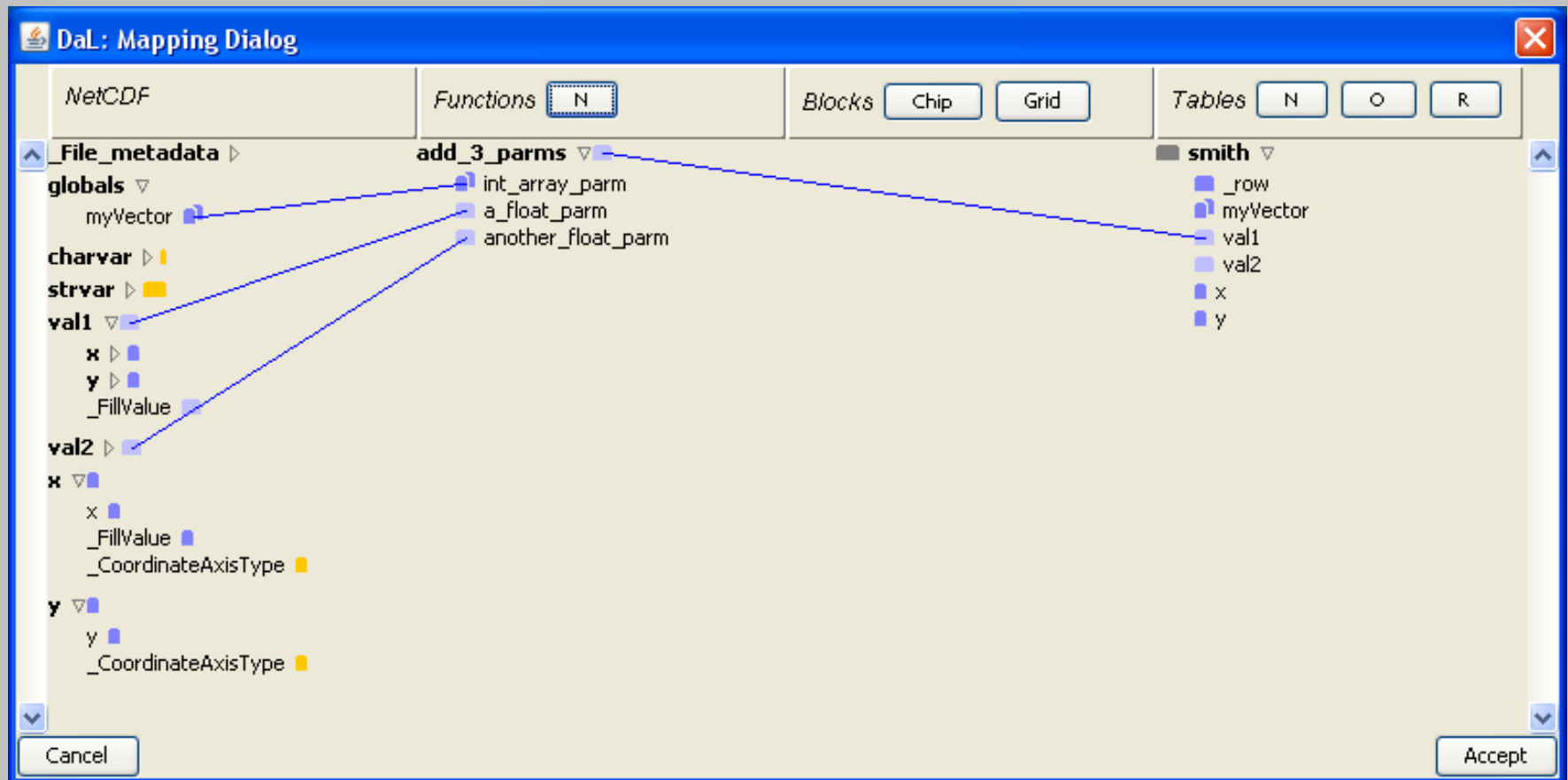
DaL - after first connection is accepted



DaL - after remaining variables are accepted

The screenshot shows the 'DaL: Mapping Dialog' window. At the top, there are tabs for 'NetCDF', 'Functions' (with a button 'N'), 'Blocks' (with buttons 'Chip' and 'Grid'), and 'Tables' (with buttons 'N', 'O', and 'R'). The main area is divided into two panes. The left pane is titled '_File_metadata' and contains a tree structure with the following items: 'globals' (containing 'myVector'), 'charvar' (containing 'val1'), 'strvar' (containing 'x' and 'y'), and 'val2' (containing 'x', 'y', and '_CoordinateAxisType'). The right pane is titled 'smith' and contains a list of variables: '_row', 'myVector', 'val1', 'val2', 'x', and 'y'. Blue lines connect the variables in the left pane to their corresponding variables in the right pane: 'myVector' to 'myVector', 'val1' to 'val1', 'x' to 'x', 'y' to 'y', and '_CoordinateAxisType' to '_row'. At the bottom of the dialog, there are 'Cancel' and 'Accept' buttons.

DaL - transforming data on the fly





Why Some Scientists Don't Use Databases

The main DBMS criticisms from a customer survey that BCS conducted in 2008 were:

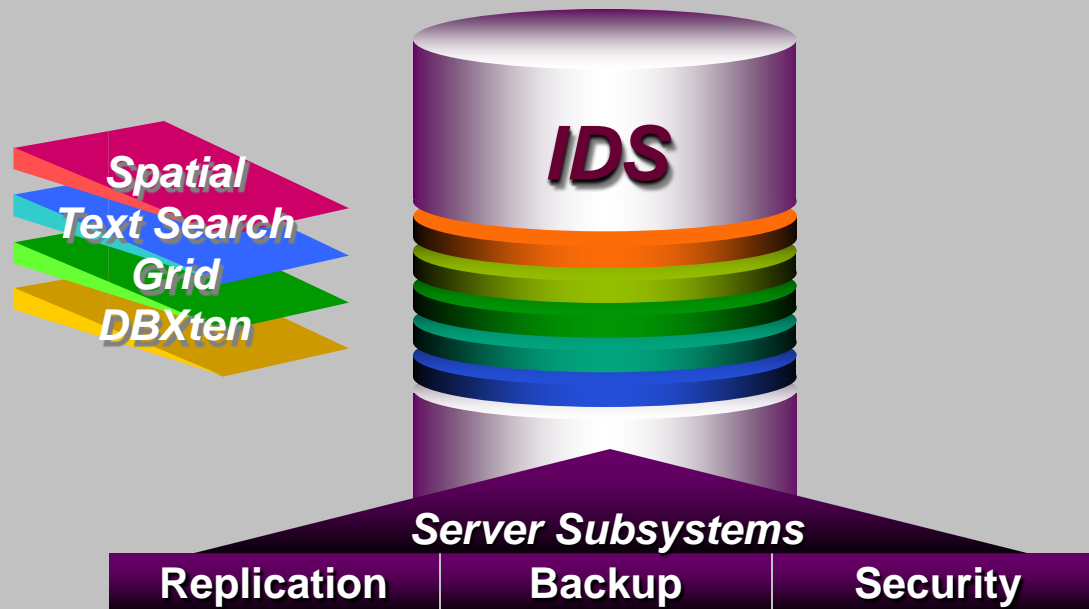
- 1. ~~“It’s not easy to get my data into a database.”~~**
- 2. “Databases don’t have all the features that I need.”**
- 3. “Databases are too slow and too big.”**



Adding Features to a DBMS

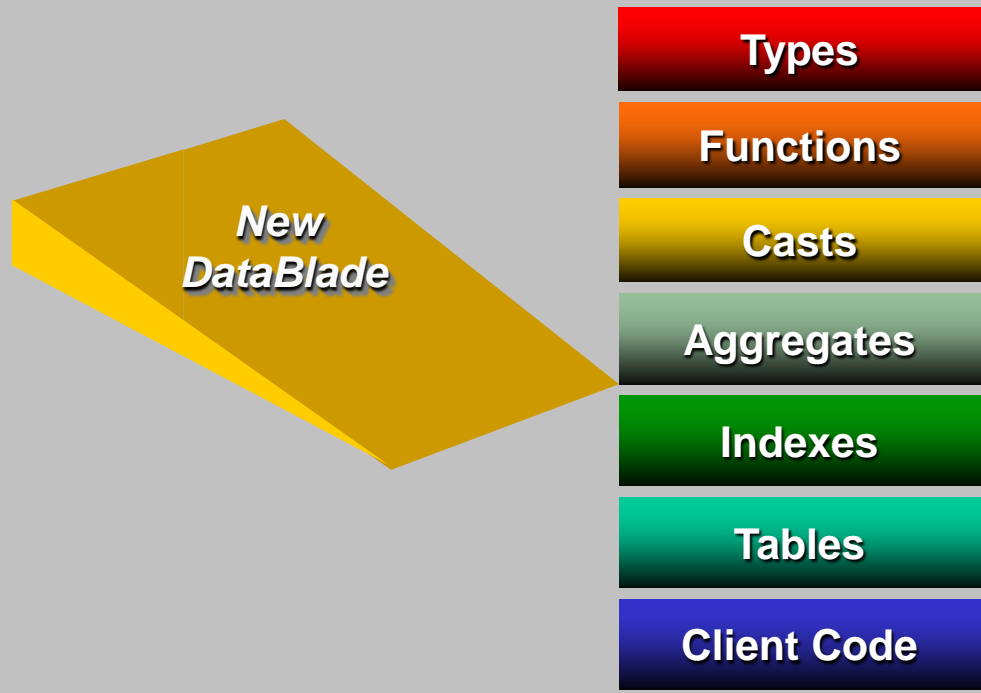
Modern databases are extensible, and so features can be added by developing database extensions. Robert Uleman of IBM will now provide background for this concept.....

Informix extensibility:
DataBlade = component (plug-in, ...)





DataBlade Elements




User-Defined Data Types (UDTs)

Normalized relational structure

employee:

name compensation location image

| | | | |
|---------------|-------------|-----------|--|
| John T. Smith | 349,876 yen | (123 256) |  |
|---------------|-------------|-----------|--|

jobs_held:

name job


| | |
|---------------|---------------|
| John T. Smith | Clerk |
| John T. Smith | Administrator |
| John T. Smith | Manager |

```
CREATE TABLE employee (  
    name                    VARCHAR(30),  
    compensation    salary_t,  
    location                point,  
    picture                 image  
);  
CREATE TABLE jobs_held (  
    name                    VARCHAR(30),  
    job                     VARCHAR(30)  
);
```

User-Defined Data Types (UDTs)

Denormalized relational structure using collection type

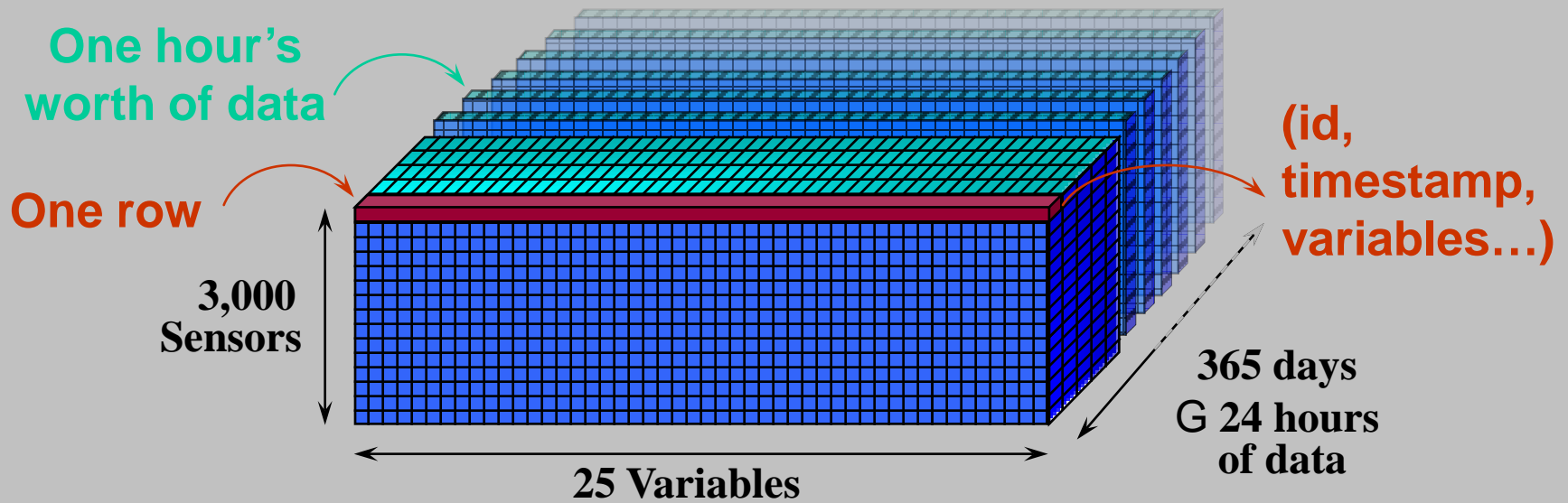
employee:

| name | compensation | location | image | jobs_held |
|---------------|--------------|-----------|---|---------------------------------|
| John T. Smith | 349,876 yen | (123 256) |  | {Clerk, Administrator, Manager} |

```
CREATE TABLE employee (  
    name          VARCHAR(30),  
    compensation  salary_t,  
    location      point,  
    picture       image,  
    jobs_held     SET(VARCHAR(30))  
);
```

Time series: relational representation

- 3,000 sensors, updated hourly, one year's data
 - 25 variables per sensor
- $3,000 \text{ G } 24 \text{ G } 365 = 26,280,000$ rows
- Rows for different sensors interspersed on disk



Time series: UDT representation

- 3,000 rows in one table
- Each TimeSeries value has all data for one sensor
- Special indexing mechanism for fast access
- Keep each sensor's data together on disk
- Server-side functions: filtering, subsampling, ...





Adding Features to a DBMS

Now that Robert has introduced the concept of a DataBlade, we'll present two specific examples of DataBlades that BCS has developed.



Gridded Data in Databases

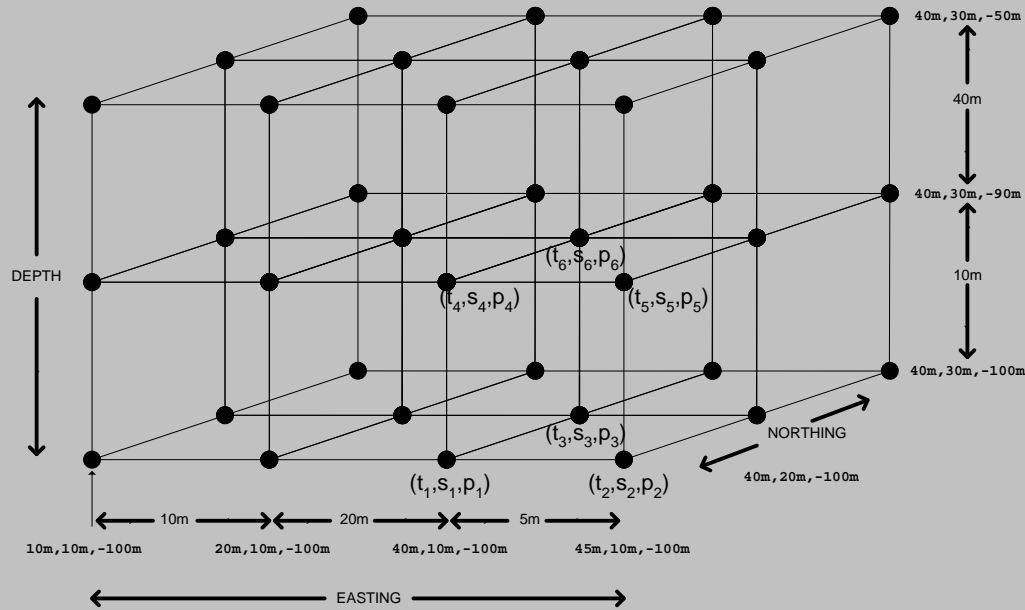
- **Gridded data occurs in meteorology, life sciences, oceanography, non-destructive testing, exploration for oil, natural gas, coal & diamonds,...**
- **Datasets range from simple, uniformly spaced grid points along a single dimension (e.g., time series) to multi-dimensional grids with many values (e.g., 4D cubes of meteorological attributes)**
- **Historically, grids were stored in flat files and then manipulated by programs that operated on these files**



The BCS Grid DataBlade

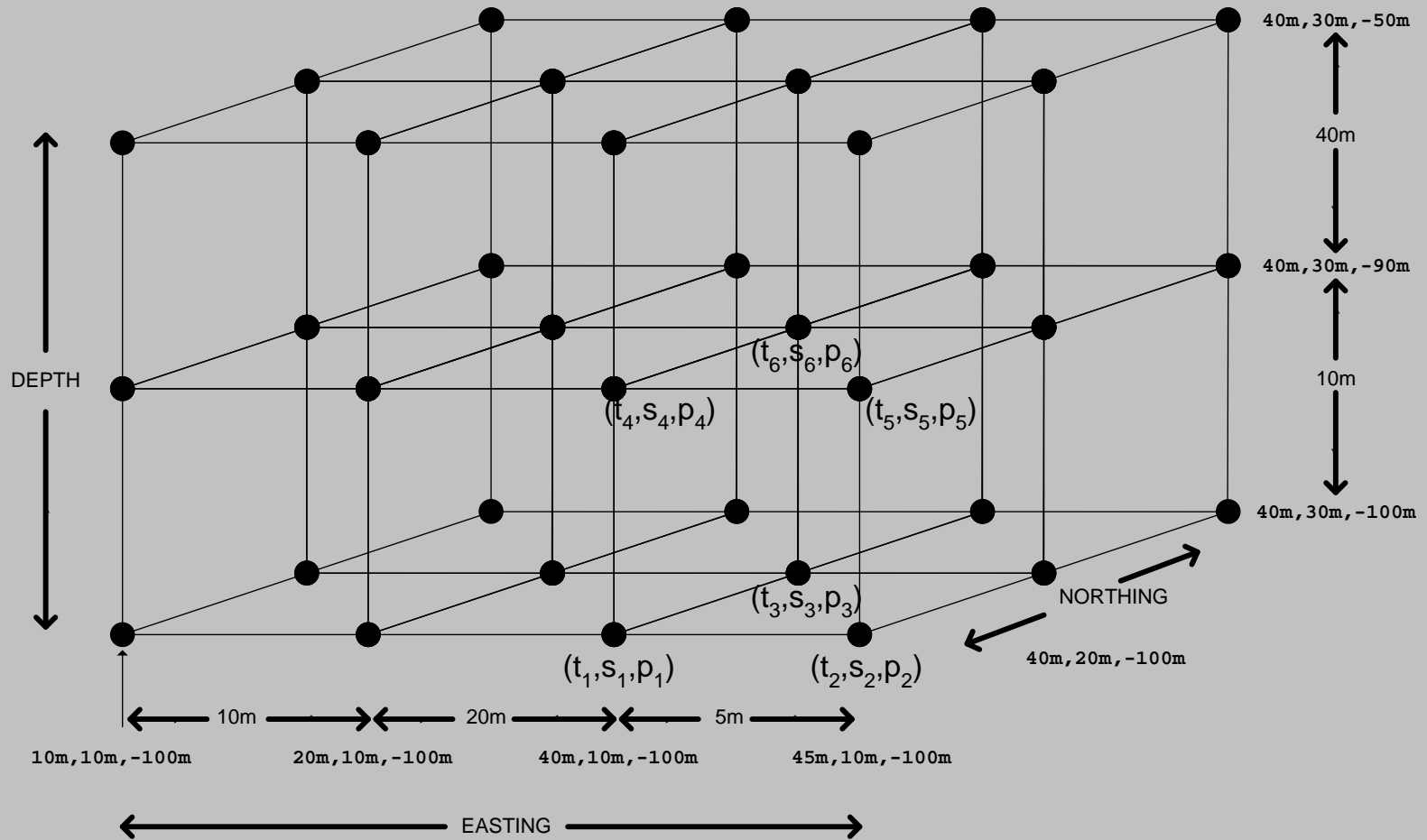
- **Handles 1D-4D grids**
- **Supports tiling**
- **Up to 50-fold increases in speed**
- **Reprojects data on the fly**
- **Can load and extract raw binary data files**
- **Provides C, Java, and SQL APIs**
- **Full user/programmer documentation freely downloadable**

BCS Grid DataBlade: Grids of Data



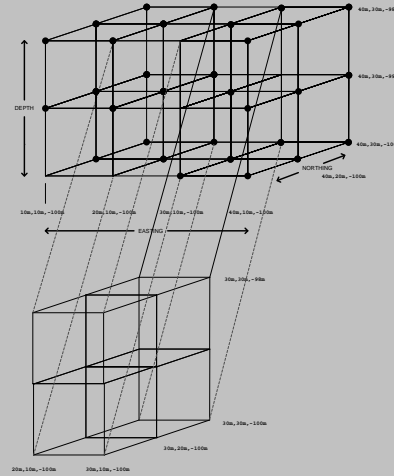
- Grids can have 1, 2, 3, or 4 dimensions
- Each grid point can store several variables
- Some grid point values can be NULL
- Grid spacing along axes can be non-uniform

BCS Grid DataBlade: Grids of Data

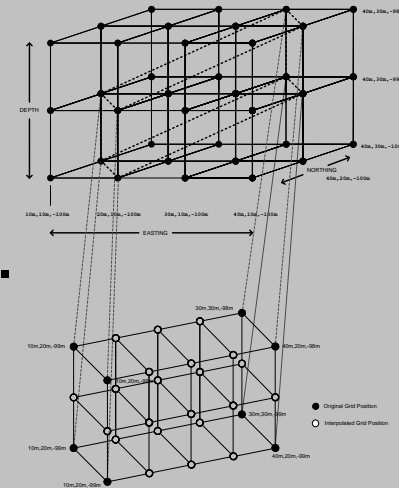


BCS Grid DataBlade: Types of Extraction

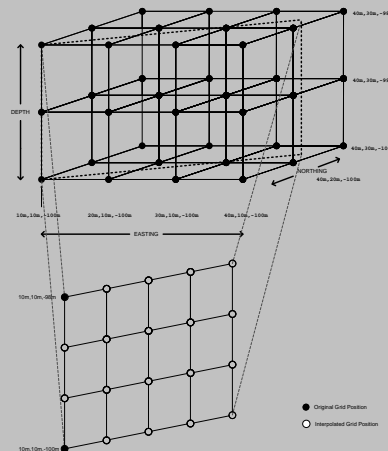
- Orthogonal



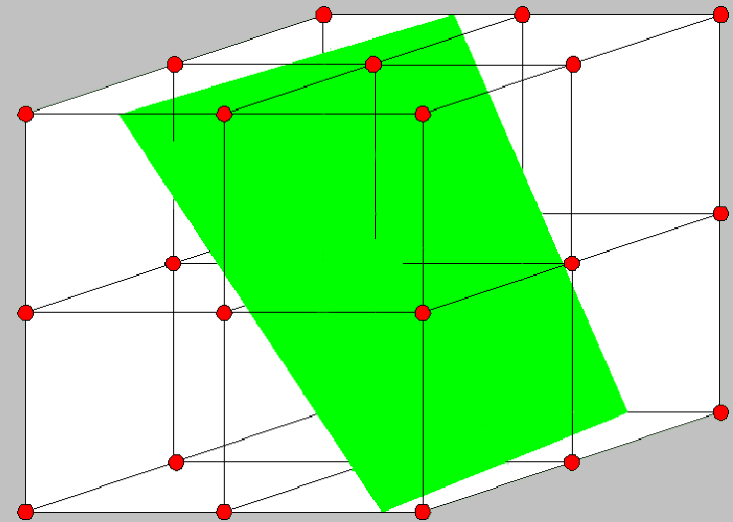
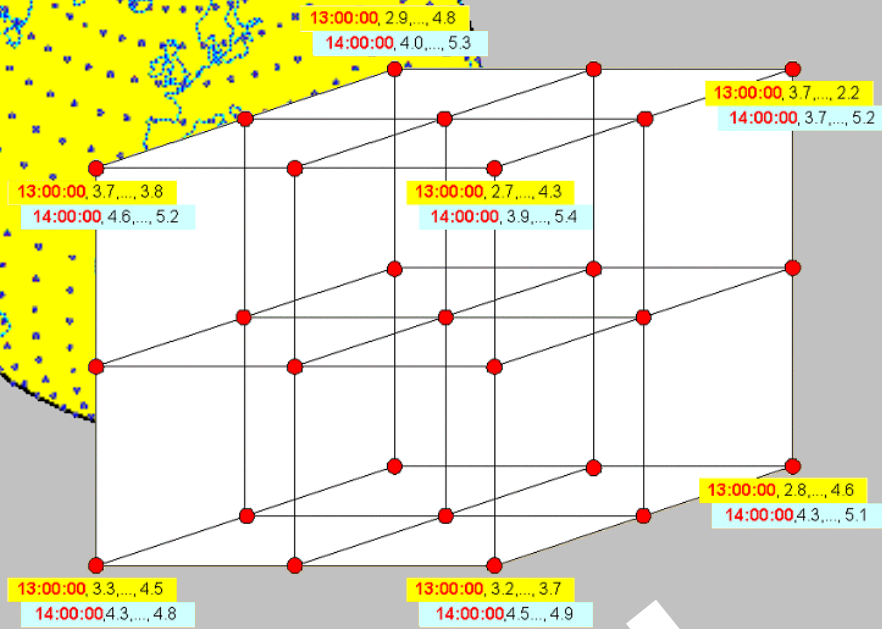
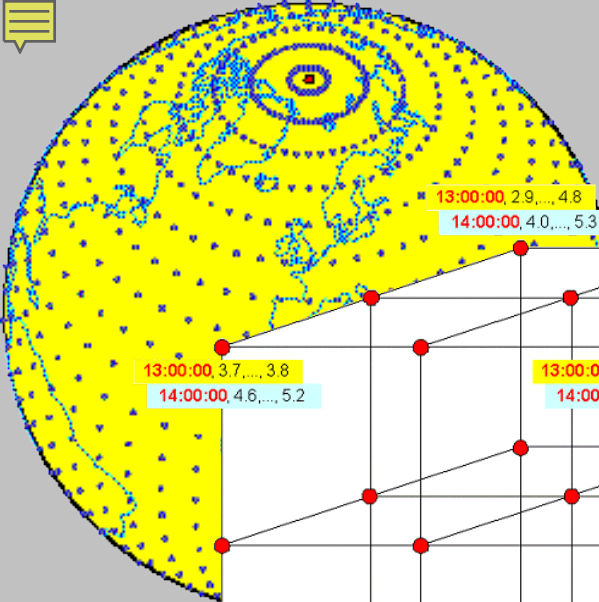
- Oblique.....



- Radial



U.S. Navy Solution



Worldwide
weather grid

Sample of
interest

DBMS

User query
Get grid
sample
of interest

SQL

Grid types
Grid functions
BCS Grid DataBlade

Used API to develop
grid types, functions
& indexes

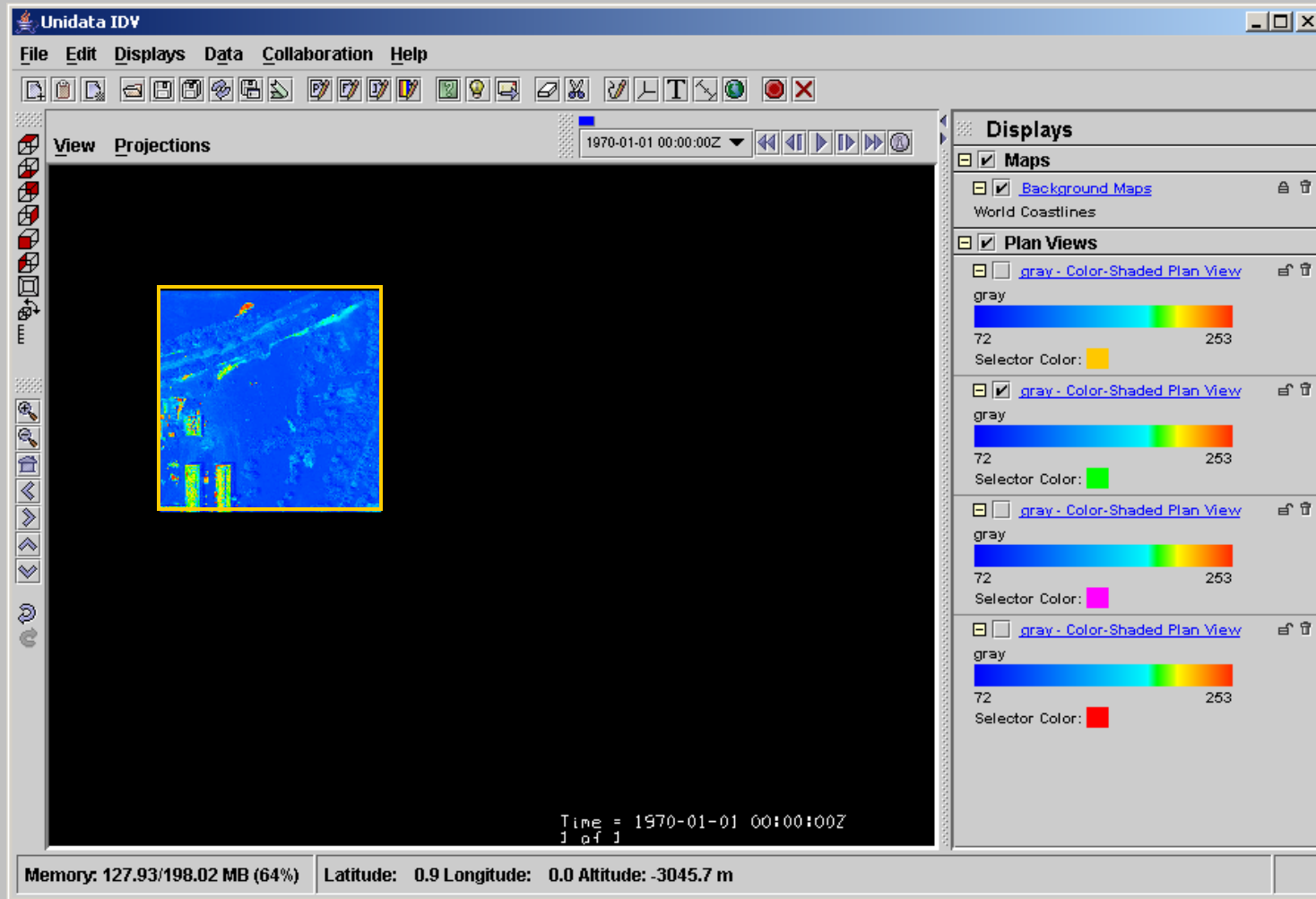
BCS

Medical Application Demo

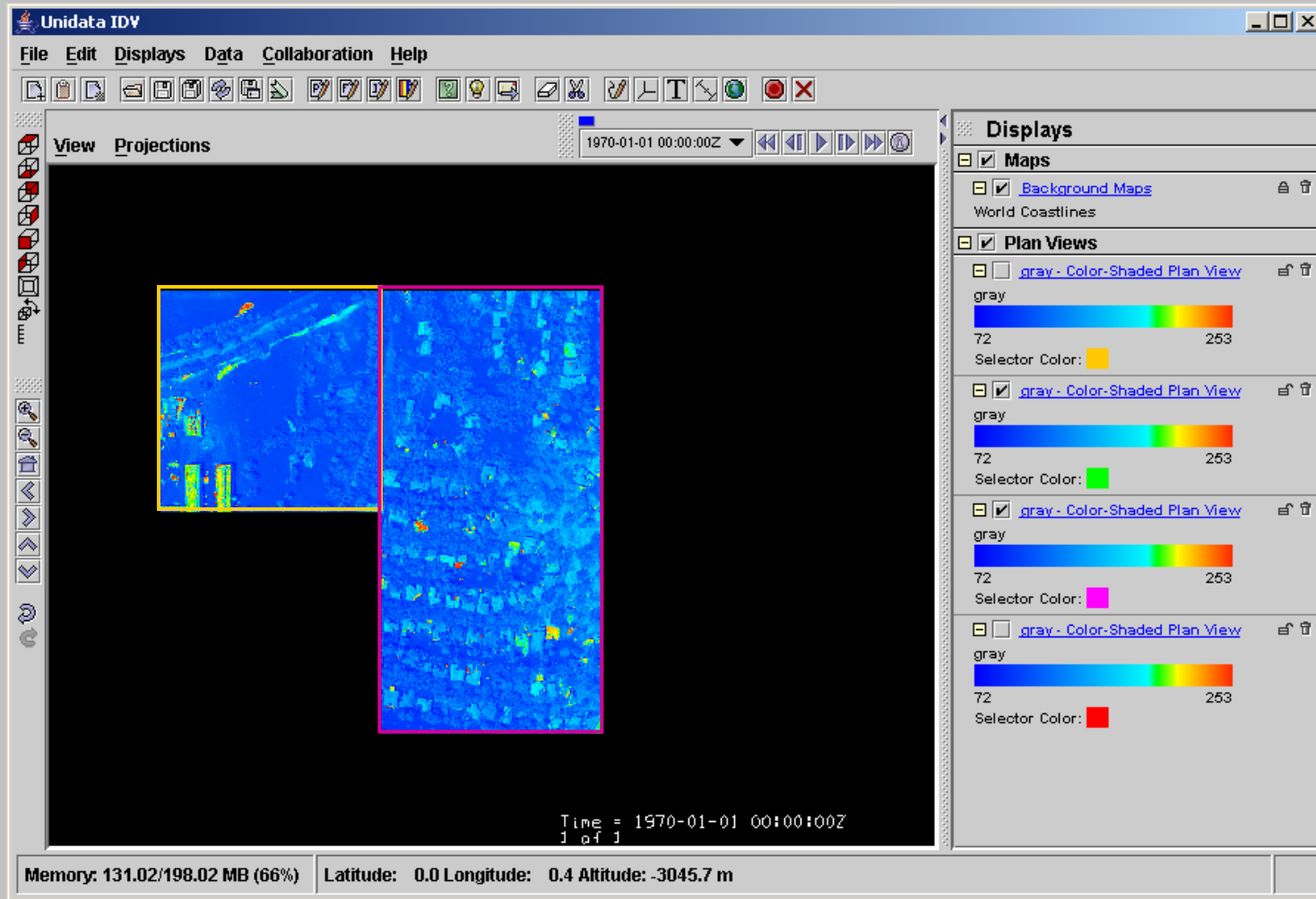
http://www.barrodale.com/grid_Demo/GridBladeApplet.html



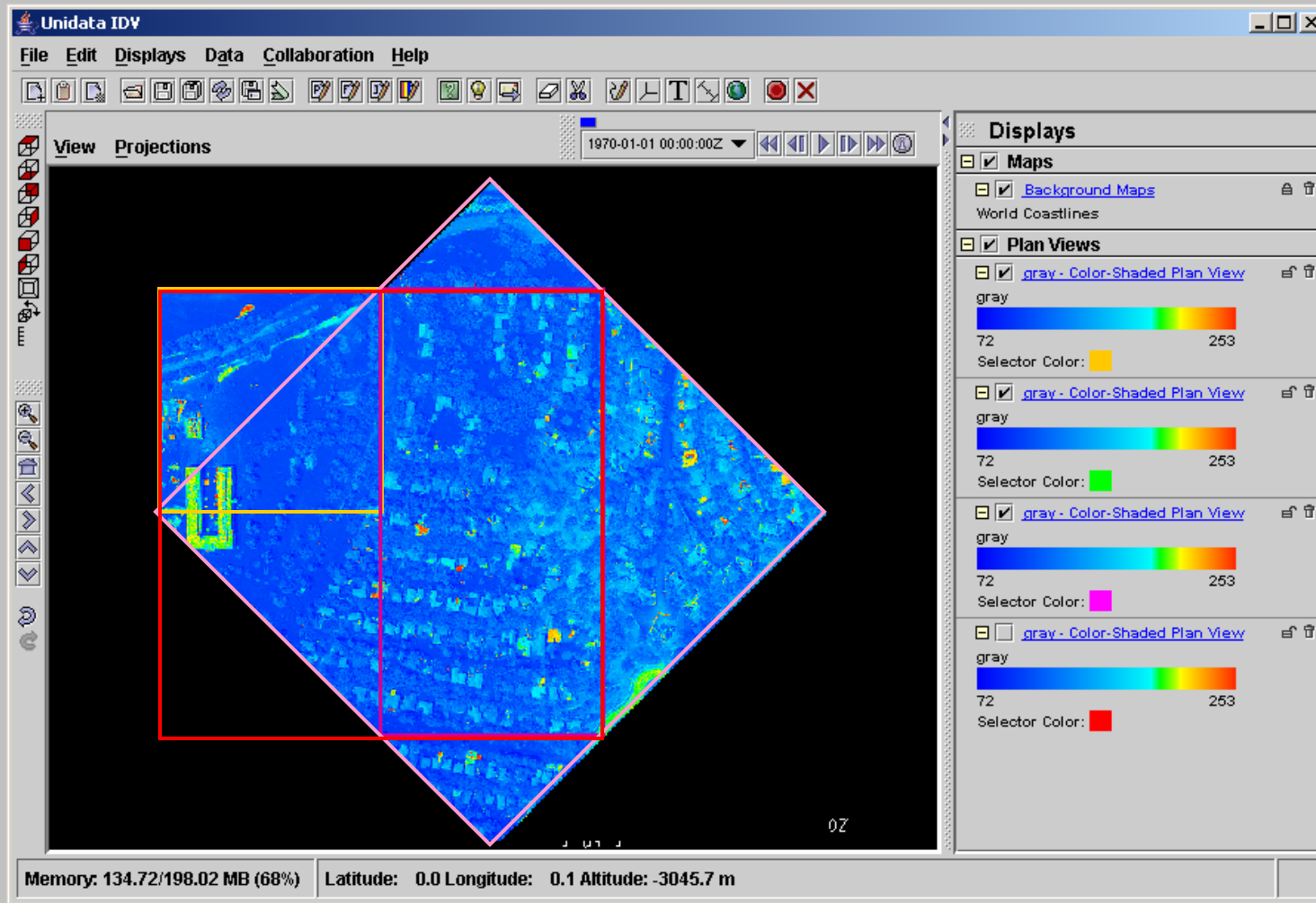
Grid Fusing: Visualized through IDV



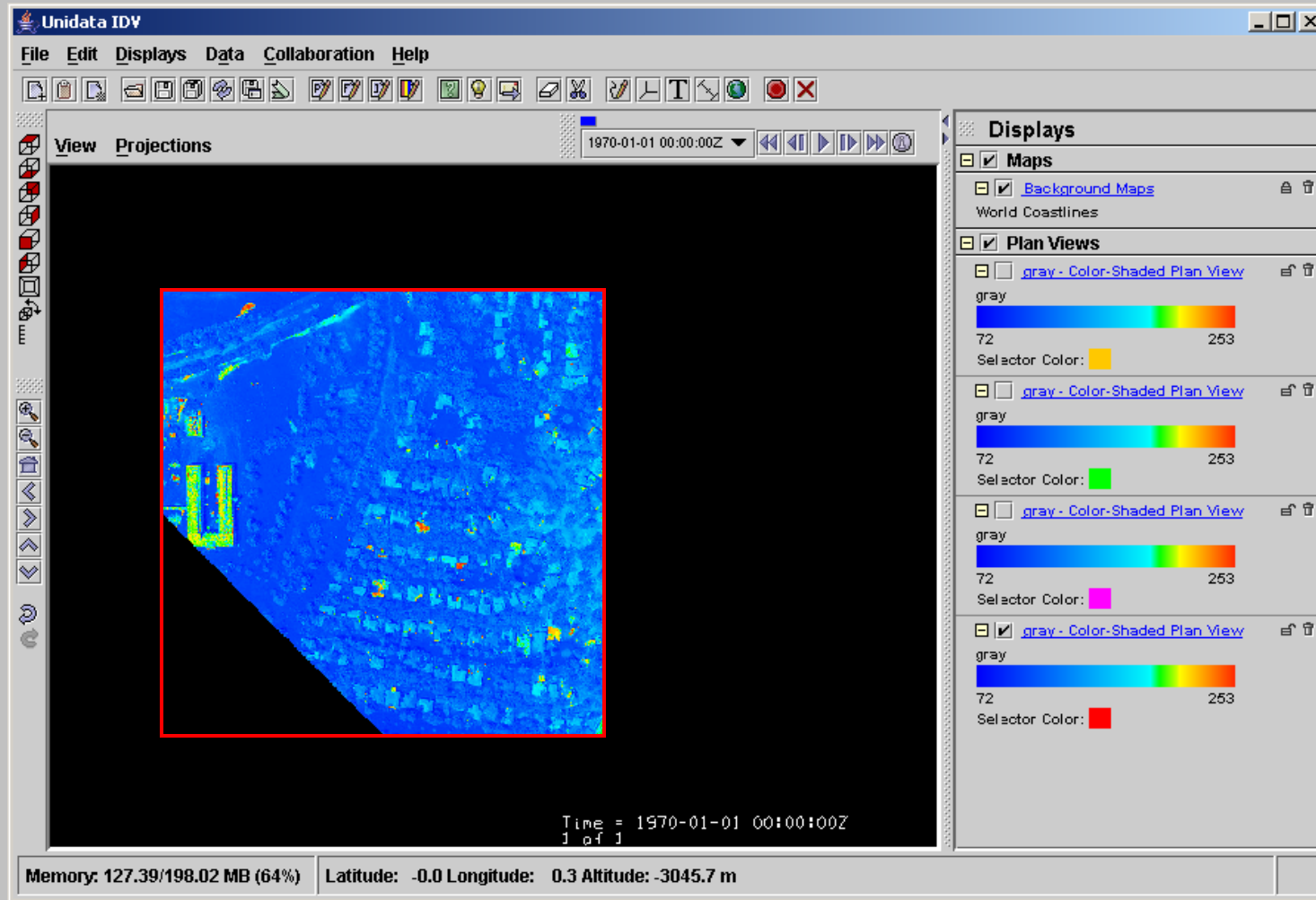
Grid Fusing: Visualized through IDV



Grid Fusing: Visualized through IDV



Grid Fusing: Visualized through IDV





SUMMARY

The BCS Grid DataBlade

... is best suited for applications where some of the following considerations apply:

- 1. The gridded data volumes are such that they can't be kept in memory**
- 2. The amount of gridded data extracted, in a particular query, is small relative to the amount stored**
- 3. The gridded data needs some form of re-sampling**

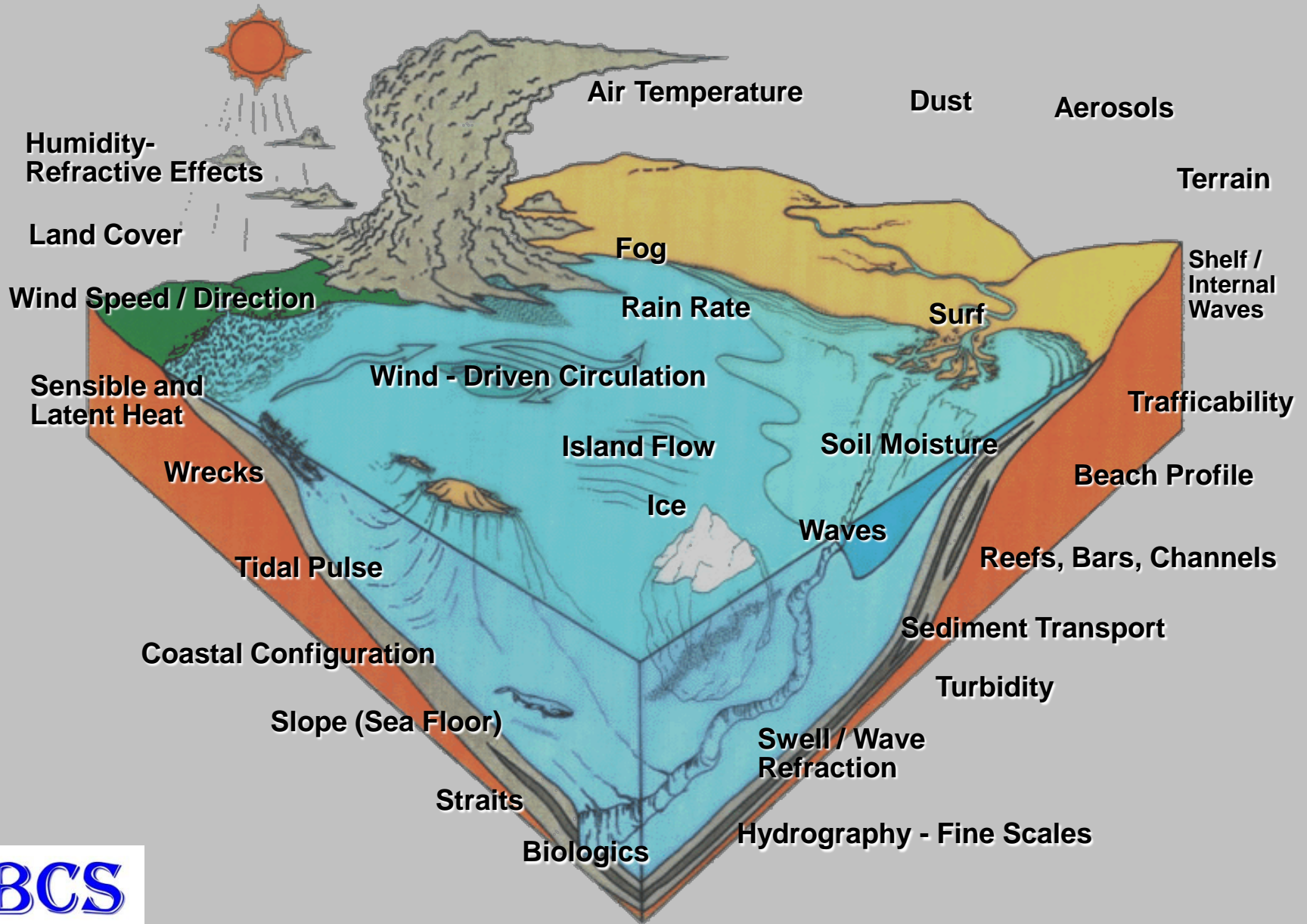


Data Sequences

Much scientific data, including Met/Ocean and geophysical data, consist of data sequences. Some examples are:

- **Meteorological/oceanographic gridded data**
- **Hydrophone/geophone array data**
- **Sea-surface temperature data from buoys**
- **Seafloor ocean observatory data**
- **GIS data – points, lines, polygons**

U.S. Navy: Environmental Attributes





Characteristics of Data Sequences

- **The data naturally occur in the form of tuples: a typical example being (x, y, z, t, pressure)**
- **The data consist of temporal and/or spatial sequences (e.g., time series, polygons)**
- **There are often strong internal correlations between the data values: e.g., grid coordinates are perfectly correlated, polygon (x, y) values are highly correlated, and pressure values are significantly correlated**



The BCS DBXten DataBlade

- **Stores tuple data**
- **Uses content-based indexing**
- **Replaces more specialized types**
- **Handles any tuple width (dimensionality)**
- **Provides compression**
- **Equipped with C, Java, and SQL APIs**
- **Full user/programmer documentation freely downloadable**



How DBXten Works

Rather than store each data tuple in a separate row, or even in external files, we take advantage of the data characteristics to:

- **Form consecutive tuples into a block**
- **Compress the separate columns of the block**
- **Store the compressed blocks in the database, along with keys for indexing**



Features of DBXten

- **Suite of a dozen data compression algorithms, applied transparently**
- **Block generation provides tiling capability**
- **Provides table-in-table storage mechanism of the data**
- **Fast and compact index generation on the stored data**
- **Fast search and retrieval capabilities using these indexes**



Why Some Scientists Don't Use Databases

The main DBMS criticisms from a customer survey that BCS conducted in 2008 were:

- 1. “It’s not easy to get my data into a database.”**
- 2. “Databases don’t have all the features that I need.”**
- 3. “Databases are too slow and too big.”**



Example of DBXten Performance

The table in the next slide shows the performance improvements obtained when DBXten was used within an IBM Informix DBMS to store a large oceanographic dataset. This dataset consisted of approximately 50,000,000 rows with 26 columns, representing NOAA sea surface temperature readings and associated data.

Example of DBXten Performance

| Task | Conventional Approach | BCS DBXten | Improvement Ratio |
|----------------------------|------------------------------|--------------------|--------------------------|
| Size of table | 15.6 GB | 1.4 GB | x 11 |
| Size of index | 6,605 MB | 6.8 MB | x 971 |
| Index creation time | 5 hours, 15 minutes | 5 seconds | x 3,780 |
| Insertion time | 1 minute, 39 seconds | 1.2 seconds | x 83 |
| Retrieval time | 14.7 seconds | 3.8 seconds | x 4 |

The results show the dramatic improvements in speed and size afforded by DBXten.



DBXten Product Development

- **DBXten is already available for IBM Informix, PostgreSQL, and Oracle**
- **We are marketing DBXten for use in meteorology, oceanography, geophysical prospecting, financial analysis, GIS, and resource management**
- **DBXten is a patent pending technology**



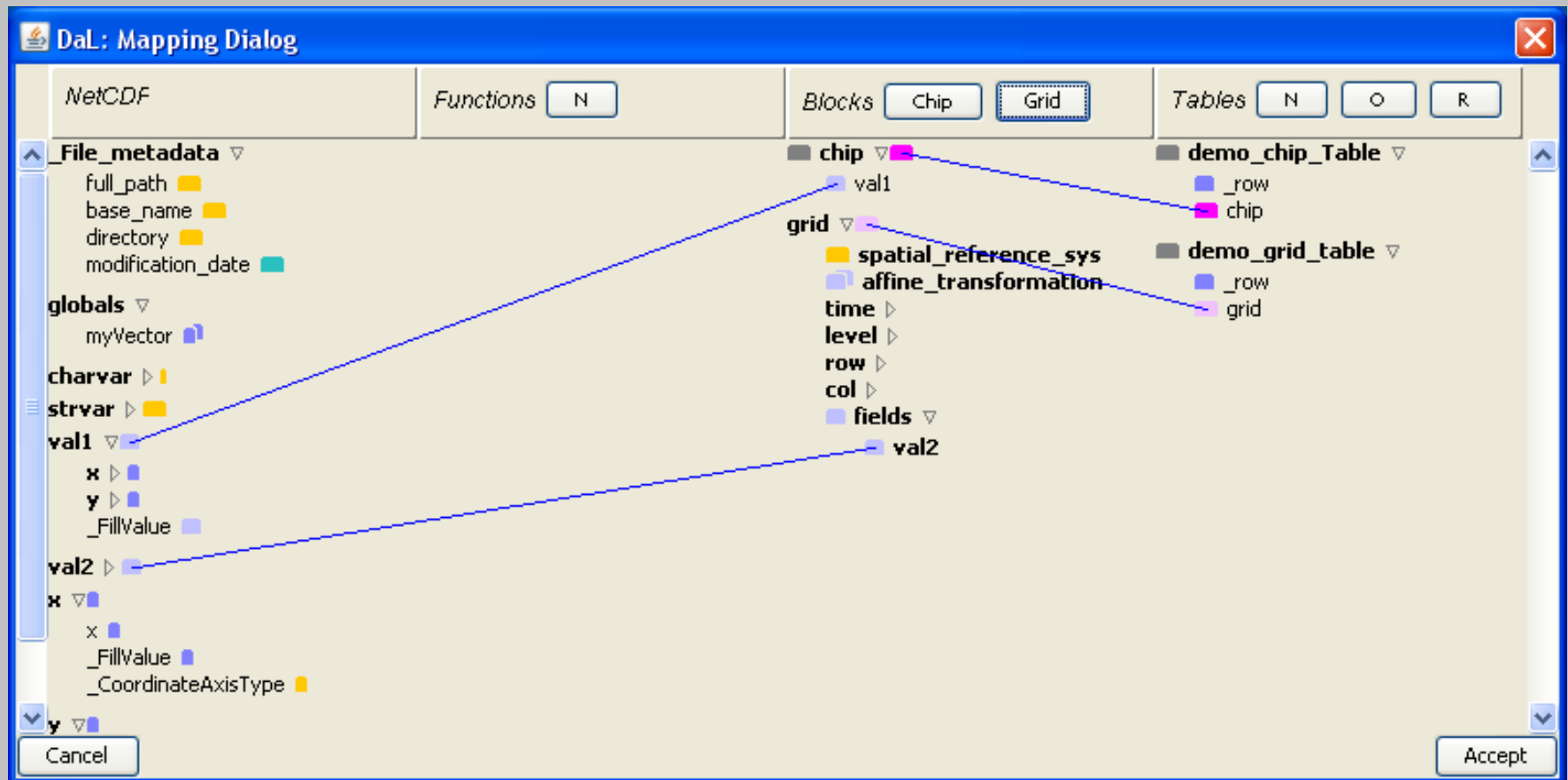
SUMMARY

The BCS DBXten DataBlade

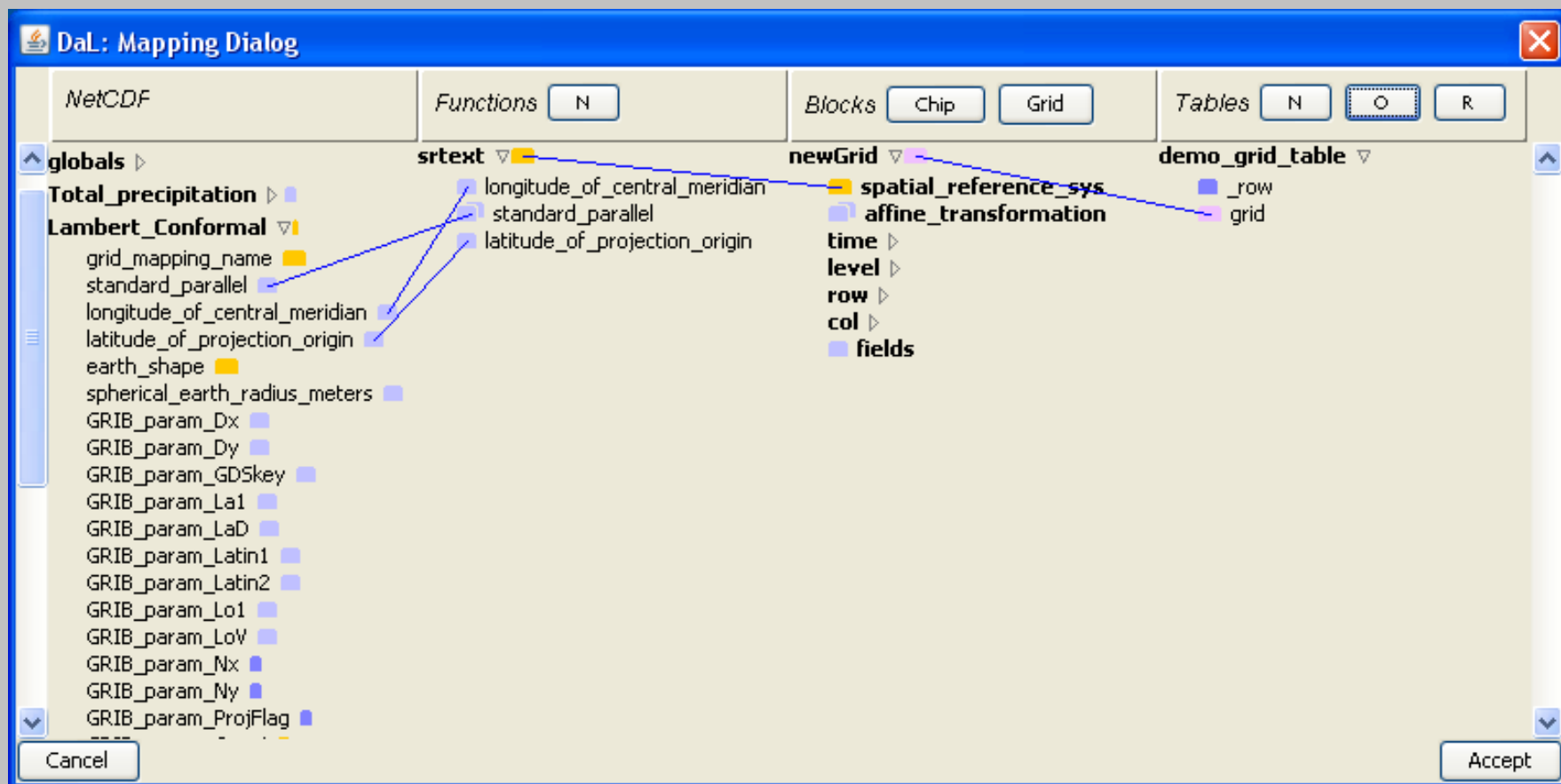
... is best suited for applications where some of the following considerations apply:

- 1. The data consists of high-volume tuple sequences**
- 2. There is a correlation between tuple rows**
- 3. Efficient content-based indexing is required**
- 4. Multiple dimensions are involved**
- 5. Compression may be required**

DaL - in use with DBXten and Grid DataBlades



DaL - defining spatial coordinate systems





We've addressed all three criticisms

1. ~~“It's not easy to get my data into a database.”~~
2. ~~“Databases don't have all the features that I need.”~~
3. ~~“Databases are too slow and too big.”~~

So, in summary.....



SUMMARY

Managing Scientific Data *Efficiently* in a DBMS

In this web conference we've shown that housing scientific data in a DBMS – *enhanced with appropriate database extensions* – can have productivity advantages (without performance losses) over managing data in flat files

We've also demonstrated the use of a free tool for putting scientific data into a DBMS – it's called DaL, and it's available from BCS (to beta testers on March 2, and for general use on March 16, 2009).



CONTACT INFORMATION:

Dr. Ian Barrodale, President

Barrodale Computing Services Ltd. (BCS)

625 Alpha Street, Suite 8

Victoria BC V8Z 1B5 Canada

(250) 412 7428

e-mail: ian@barrodale.com

For more information about BCS projects, experience, and capabilities, please visit:

<http://www.barrodale.com>

BCS