Virtual Data Language: A Typed Workflow Notation for Diversely Structured Scientific Data

Yong Zhao[1], Michael Wilde[23], Ian Foster[123]

[1]Department of Computer Science, University of Chicago
[2]Computational Institute, University of Chicago
[3]Division of MCS, Argonne National Laboratory

DSL Workshop 2006, 02 June 2006

Outline

- Motivation
- Typing System
- XDTM (XML Dataset Typing and Mapping)
- Virtual Data Language
- fMRI Use Case
- Conclusion

The Broad Picture – Data and Grid

- Data analysis turns into data integration
  - The need to discover, access, explore, analyze diverse distributed data sources
- Science as collaborative workflow
  - The need to organize, archive, reuse, explain, and schedule scientific workflows
- Virtual data as a unifying concept
  - Integrated view over data, programs, and computations

Challenges

- Deluge of data
- Heterogeneity
  - Diversely structured data storage and formats
  - Metadata encoded in ad hoc ways
- Geographic and political distribution
  - Different administrative domains
  - Different access protocols and policies
- Collaboration within/across large, dynamic communities
  - Negotiation and sharing of resources

“Messy” Scientific Data

- Heterogeneous storage format and access protocol
  - Logically identical dataset can be stored in
    - Textual File (e.g. CSV), binary file, spreadsheet
  - Data available from
    - Filesystem, database, HTTP, WebDAV, etc...
- Metadata encoded in directory and file names
  - A fMRI volume is composed of an image file and a header file with the same prefix.
- Format dependency hinders program and workflow reuse

But... Data is often Logically Structured

- Scientific data often maintain hierarchical structure
- A common practice is to select a set of data items and apply a transformation to each individual item
- A nested approach of such iterations could scale up to millions of objects

Introducing a Typing System

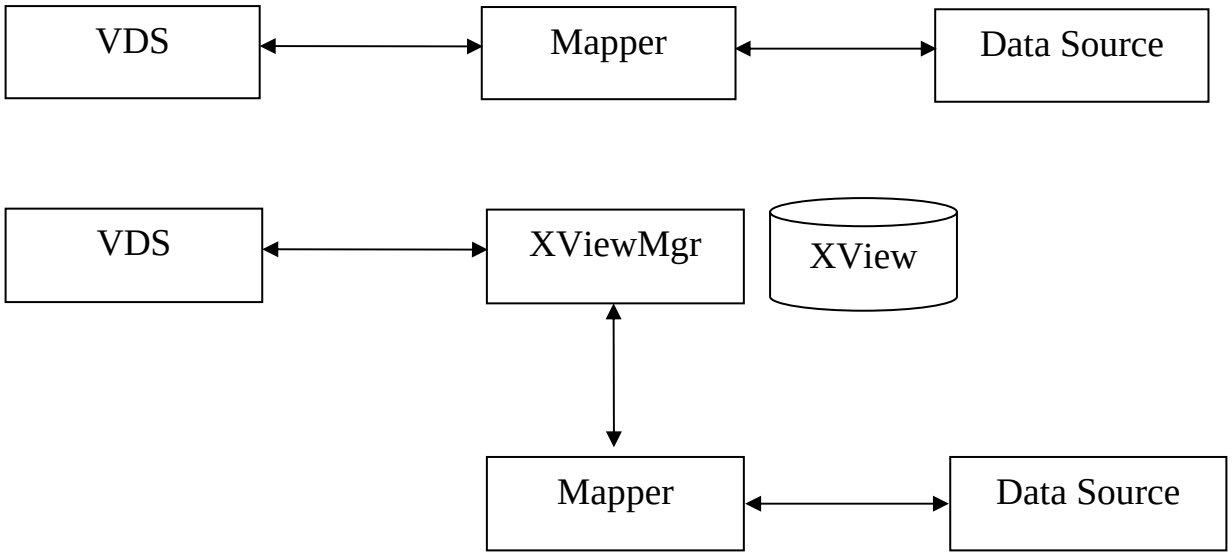
- Describe logical data structures as types
- Define procedures in terms of typed datasets, use such procedures on different physical representations
- Compose workflows from typed procedures
- Benefits
  - Type checking
  - Dataset selection and iteration
  - Discovery by types
  - Dynamic binding
  - Type conversion

XDTM

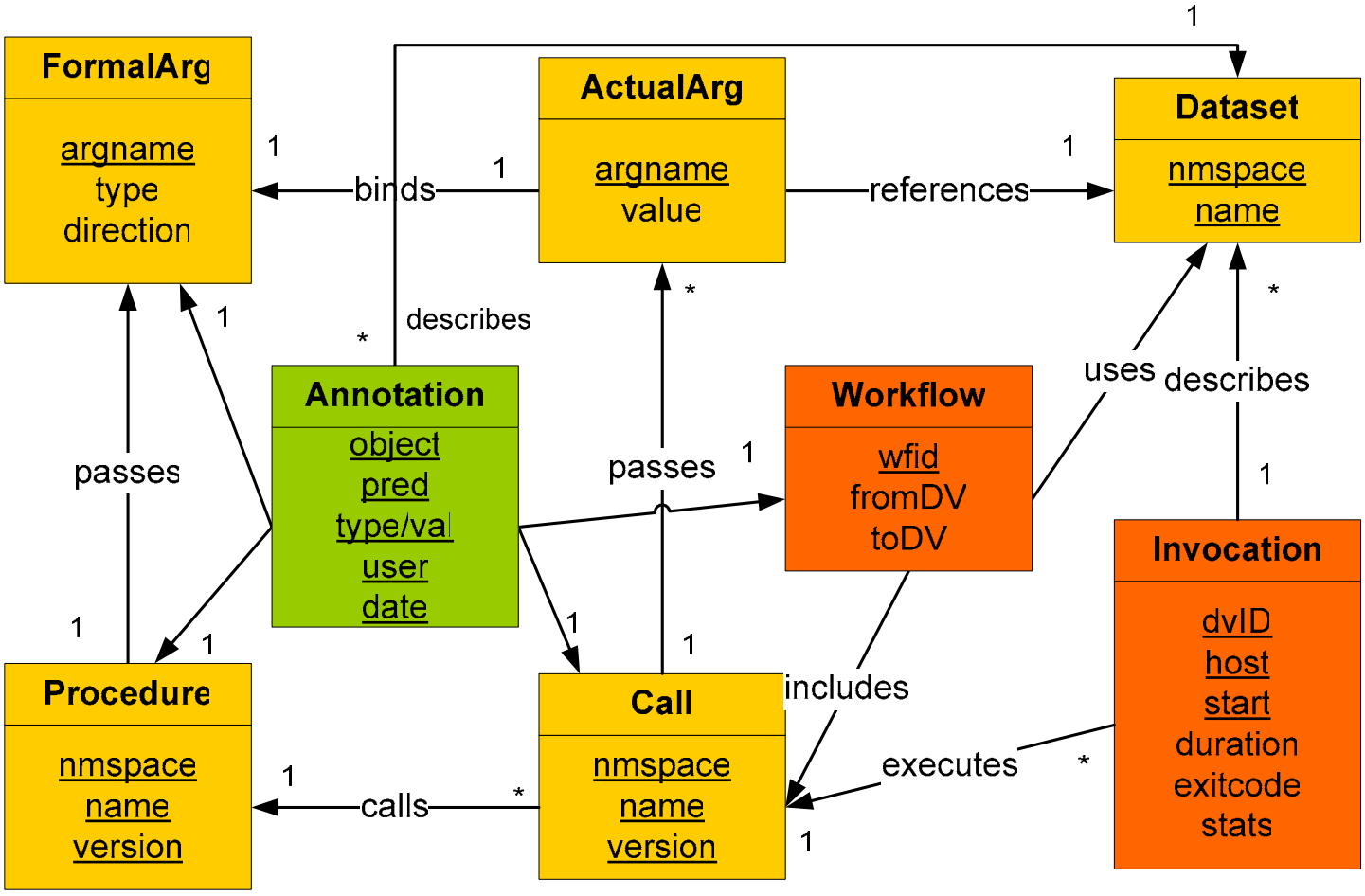
- XML Dataset Typing and Mapping
- Separates logical structure from physical representations
- Logical structure described by XML Schema
  - Primitive scalar types: int, float, string, date …
  - Complex types (structs and arrays)
- Mapping descriptor
  - How dataset elements are mapped to physical representations
  - External parameters (e. g. location)
- XPath for dataset selection

Mapping

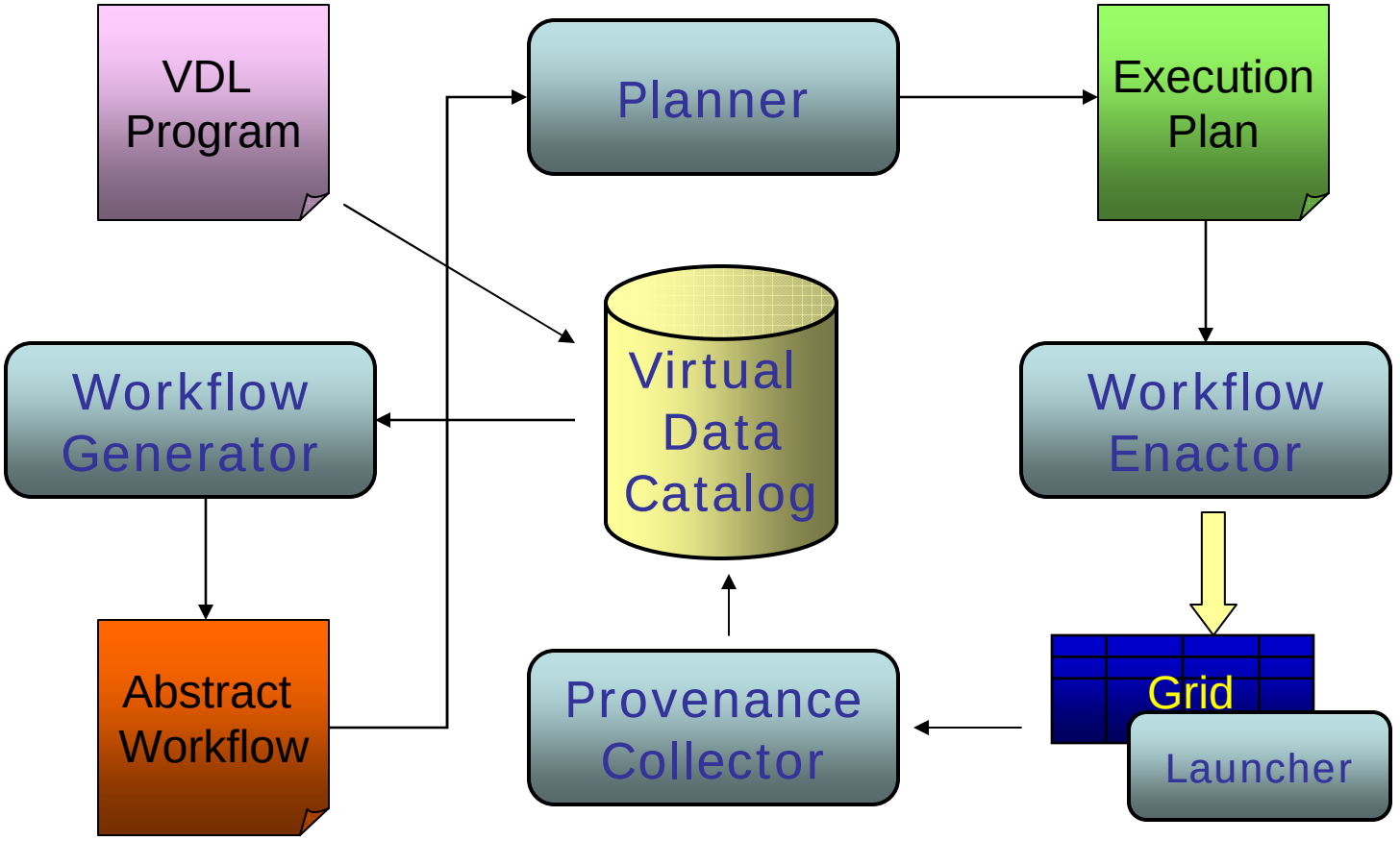
- Define a common mapping interface
  - Initialize, read, create, write, close
- Data providers implement the interface
  - Responsible for data access details
- XView maintains cached logical datasets



Virtual Data Schema



Virtual Data System



Use Case – fMRI Data

Logical Structure

DBIC Archive
- Study #1
  - Group #1
    - Subject #1
      - Anatomy
        - high-res volume
      - Functional Runs
        - run #1
          - volume #001
          - ...
          - volume #275
        - ...
        - run #5
          - volume #001
          - ...
        - snrun #...
        - …
  - Group #5
  - ...
- Study #...

Physical Representation

DBIC Archive
- Study_2004.0521.hgd
  - Group_1
    - Subject_2004.e024
      - volume_anat.img
      - volume_anat.hdr
      - bold1_001.img
      - bold1_001.hdr
      - ...
      - bold1_275.img
      - bold1_275.hdr
      - ...
      - bold5_001.img
      - ...
      - snrbold*_*
      - air*
      - ...
  - Group_5
    - ...
- Study ...

Type Definitions in VDL

```
type Image {};

type Header {};

type Volume {
        Image img;
        Header hdr;
}

type Anat Volume;

type Warp {};

type NormAnat {
        Anat aVol;
        Warp aWarp;
        Volume nHires;
}
```

```
type Run {
        Volume v [ ];
}

type Subject {
        Anat anat;
        Run run [ ];
        Run snrun [ ];
}

type Group { Subject s[ ]; }

type Study { Group g[ ]; }
```

Part of fMRI AIRSN (Spatial Normalization) Workflow

Type Definitions in XML Schema

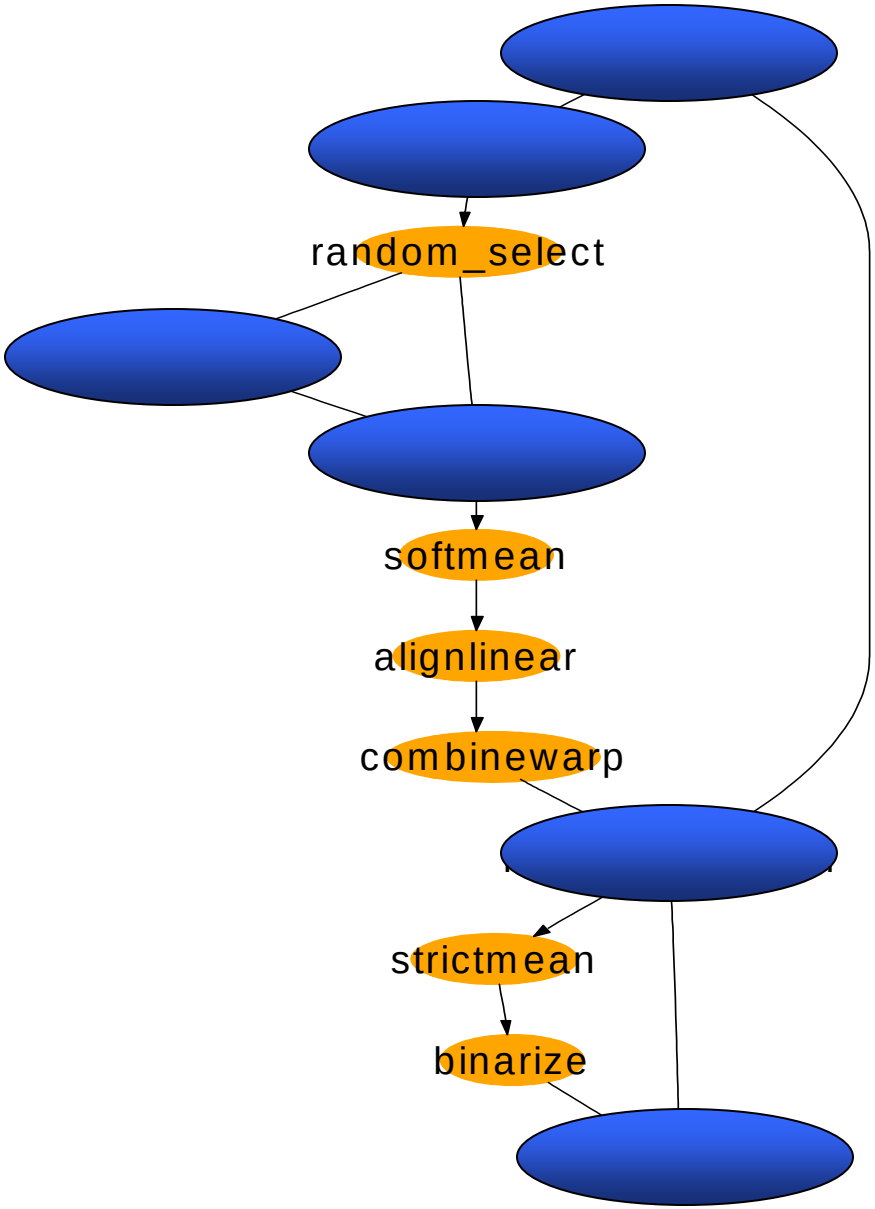
```
<xs:schema
        targetNamespace="http://www.fmri.org/schema/airsn.xsd"
        xmlns="http://www.fmri.org/schema/airsn.xsd"
        xmlns:xs="http://www.w3.org/2001/XMLSchema">

        <xs:simpleType name="Image"/>
        <xs:simpleType name="Header"/>

        <xs:complexType name="Volume">
            <xs:sequence>
                <xs:element name="img" type="Image"/>
                <xs:element name="hdr" type="Header"/>
            </xs:sequence>
        </xs:complexType>

        <xs:complexType name="Run">
            <xs:sequence minOccurs="0" maxOccurs="unbounded">
                <xs:element name="v" type="Volume"/>
             </xs:sequence>
        </xs:complexType>
</xs:schema>
```

Procedure Definition in VDL

```
(Run snr) functional( Run r, NormAnat a, Air shrink ) {
    Run yroRun = reorientRun( r , "y" );
    Run roRun = reorientRun( yroRun , "x" );
    Volume std = roRun[0];
    Run rndr = random_select( roRun, .1 ); //10% sample
    AirVector rndAirVec = align_linearRun( rndr, std, 12, 1000, 1000, [81,3,3]
    );
    Run reslicedRndr = resliceRun( rndr, rndAirVec, "o", "k");
    Volume meanRand = softmean(reslicedRndr, "y", null );
    Air mnQAAir = alignlinear( a.nHires, meanRand, 6, 1000, 4, [81,3,3] );
    Volume mnQA = reslice( meanRand, mnQAAir, "o", "k“ );
    Warp boldNormWarp = combinewarp( shrink, a.aWarp, mnQAAir );
    Run nr = reslice_warp_run( boldNormWarp, roRun );
    Volume meanAll = strictmean ( nr, "y", null )
    Volume boldMask = binarize( meanAll, "y" );
    snr = gsmoothRun( nr, boldMask, 6, 6, 6 );
}
```

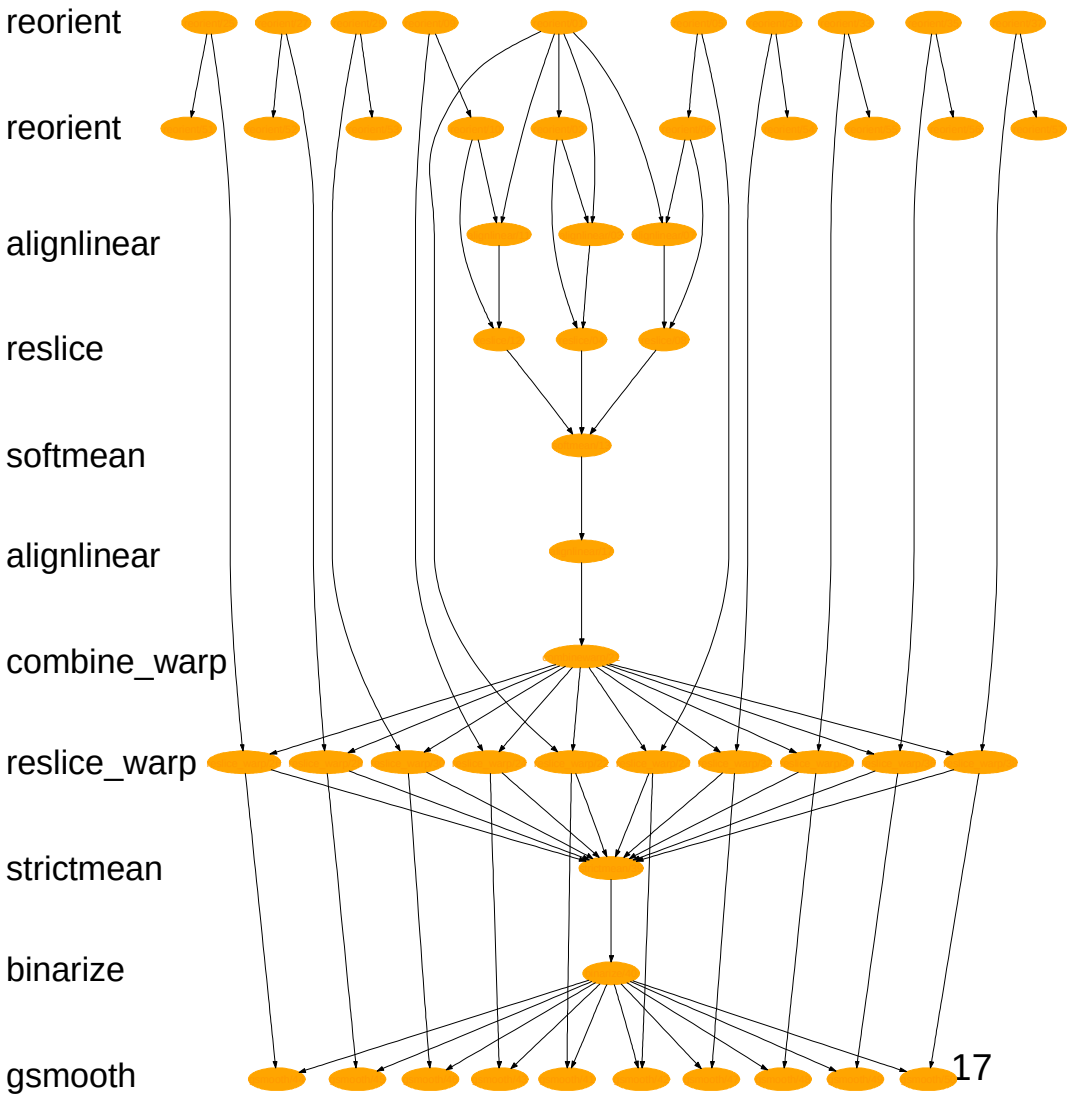
Dataset Iteration

- Functional analysis expressed in typed datasets
- Iterate over each *volume* in a *run*



Expanded Execution Plan

- Datasets dynamically instantiated from data sources by mappers



Code Size Comparison

Lines of code with different workflow encoding

Workflow	Script	Generator	VDL
GENATLAS1	49	72	6
GENATLAS2	97	135	10
FILM1	63	134	17
FEAT	84	191	13
AIRSN	215	~400	37

Conclusion

- XDTM provides the data model for separation of logical structure from physical representations
- VDL allows workflow composition and dataset iteration based on typed signature
- fMRI use case proves effectiveness and productivity gain

For More Information

- GriPhyN
  - http://www.griphyn.org/
- VDS
  - http://www.griphyn.org/vds
- Publications
  - http://people.cs.uchicago.edu/~yongzh