An XML-based Data Access Interface for Image Analysis and Visualization Software

Syam Gadde, Charles R. Michelich, James Voyvodic

Duke-UNC Brain Imaging and Analysis Center, Durham, NC, USA
Author contact: gadde@biac.duke.edu

1 INTRODUCTION

Constructing a robust image analysis pipeline from several, often disparate, software elements is not a trivial process. Many valuable software tools have arbitrary data format requirements. Combining such software tools in analysis requires data conversion steps that may slow down processing and introduce errors.

We have defined a general data access interface in the form of a structured text file, written in XML (Extensible Markup Language). This XML file provides instructions on how to extract image data from files in any uncompressed data format, without explicit conversion of data. This interface supports data of arbitrary dimensionality, data type, and byte order, and is applicable to most MR image formats, physiological and stimulus data.

2 GOALS

COMPONENT REUSE

Processing tools that read and write data in multiple formats typically share the following pipeline:

- ❖ **Assimilate:** Read and convert data from input format into common internal data structure
- ❖ **Compute:** Perform operations on internal data structure
- ❖ **Disseminate:** Convert and write data from internal data structure to output format

Supporting multiple input and output data formats requires individual reader and writer modules for each supported data format (Fig. 1). Moreover, these modules must be re-implemented in each processing tool. One goal of our work is to implement an infrastructure that minimizes the number of required processing pipeline components and promotes the reuse of such components. We accomplish this by moving the *assimilation* component out of the processing tool and storing the results externally in a human- and machine-readable XML file (Fig. 2). Also, by *disseminating* XML files with the output, processing tools make their results available to other XML-aware applications.

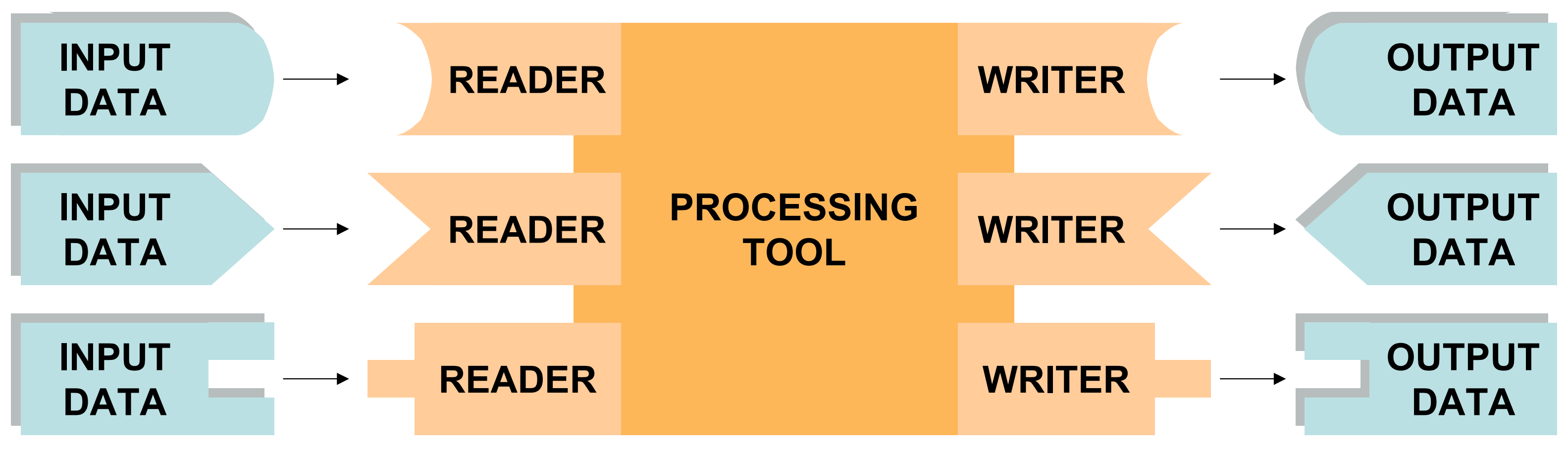

Figure 1: Supporting multiple data formats requires individual readers/writers for each data format. Supporting future formats will require extending the tool itself to add new reader and writer modules.

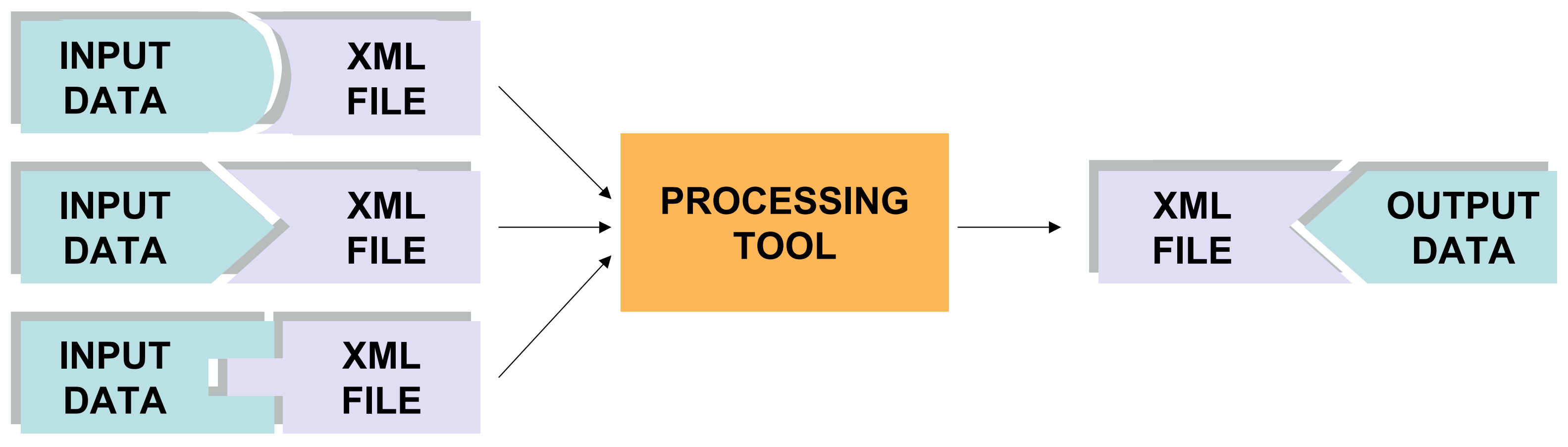

Figure 2: In this case, XML files are the input to the XML-aware processing tool. The XML files provide a common interface to the data in the input files, regardless of their format. This processing tool now can automatically read any future data formats provided that there is a way to encapsulate those formats using the XML-based data description framework. The output XML file will also be readable by other XML-aware tools.

ROBUSTNESS

Many data formats exist and will continue to be developed. Our processing pipeline must be robust to the introduction of new data formats. By placing the responsibility for understanding and supporting new data formats only with the *assimilation* tool, we remove this burden from individual processing tools. These XML-aware tools, unmodified, can then read the data using the resulting XML files.

TRANSPARENCY

Explicit conversions of data between formats almost always compromises the quality of the data or metadata (i.e. descriptive fields stored with the data), because format A may not support the same data/metadata types supported by format B. Moreover, legacy/proprietary software may require data to be in the original format to function properly. It is also good practice to keep data in its original form to enable future reprocessing.

For these reasons, we create XML files as lightweight "wrappers" for the original data, without the need for any explicit data format conversion. The original data remains unmodified and the XML file is placed alongside the data it encapsulates, pointing to the raw data files by reference when necessary. Indeed, the original data is integral to this approach – the XML file cannot be used on its own except as a summary of extracted metadata elements.

ACCESSIBILITY

The XML markup language provides several advantages over simpler plain text formats. Because XML is merely structured text, it is easily parsed and used as a common access interface for image dimensions, subject information, acquisition parameters, or transformations between image and world coordinate systems. It is also human-readable, obviating the need for special tools to perform occasional validity checks. The XML structure itself is hierarchical; this, coupled with the use of XML namespaces, facilitates both the incorporation of this data within other XML documents and the integration of additional metadata modules.

3 DESIGN

Our typical automated processing pipeline incorporates both XML-aware software and legacy software. Such legacy software is not easily modified and may require conversions to particular file formats. For these components we use the XML file to carry over image metadata we might otherwise lose when converting from a richly annotated data format (e.g. DICOM) to one that is sparsely annotated (e.g. Analyze 7.5) (Fig. 3).

The core of our XML file is the *data record* (Fig. 4), which describes the original data in terms of dimensionality, element type, byte order, data file names, position of data within files, etc. This data record is constructed by a tool that understands the layout of the original data format, and translates it into the XML-based description language. The tool may also extract key metadata elements (e.g. acquisition parameters for MR data files) and store them in the XML file (Fig. 5).

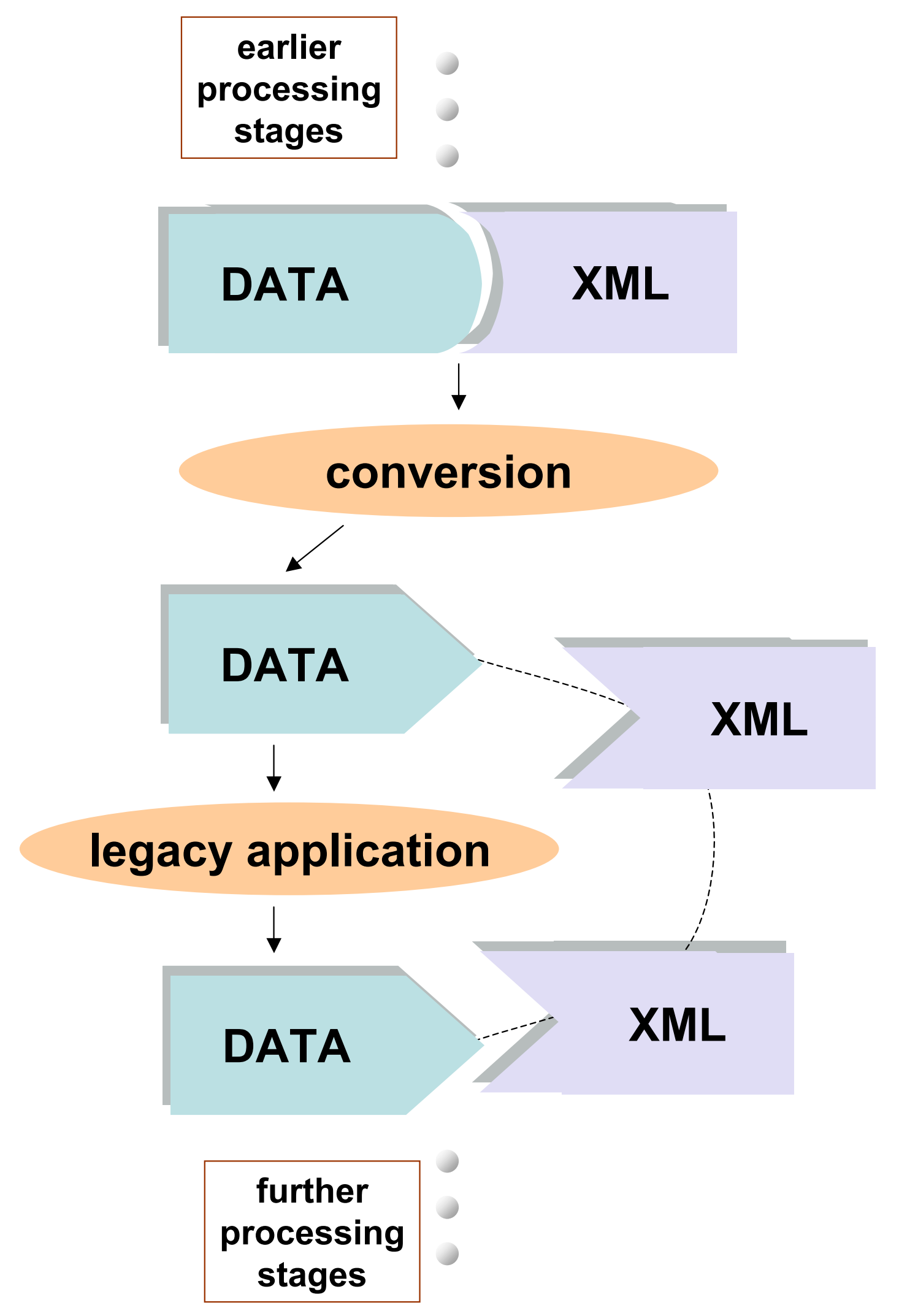

Figure 3: XML files can be used to protect metadata when legacy applications demand conversions to sparsely-annotated data formats.

```
•••
<datarec type="image">
    <!-- Orientation is oblique axial -->
    <dimension type="x">
        <units>mm</units>
        <size>64</size>
        <origin>118.125017</origin>
        <gap>0</gap>
        <spacing>3.75</spacing>
        <direction>-1 -0 0</direction>
    </dimension>
    <dimension type="y">
        <units>mm</units>
        <size>64</size>
        <origin>98.760873875</origin>
        <gap>0</gap>
        <spacing>3.75</spacing>
        <direction>-0 -0.997 -0.078</direction>
    </dimension>
    <dimension type="z">
        <units>mm</units>
        <size>2</size>
        <origin>55.25402975</origin>
        <gap>0</gap>
        <spacing>3.8</spacing>
        <direction>0 0.078 -0.997</direction>
    </dimension>
    <byteorder>lsbfirst</byteorder>
    <elementtype>int16</elementtype>
    <filename>scan_00001.dcm</filename>
    <fileoffset>9512</fileoffset>
    <filerecordsize>8192</filerecordsize>
    <filename>scan_00002.dcm</filename>
    <fileoffset>9510</fileoffset>
    <filerecordsize>8192</filerecordsize>
</datarec>
•••
```

Figure 4: An example of a *data record*, which stores enough metadata to read and display the data in the raw data files.

	size	gap	spacing	orientation (RAS)		
X	64	0	3.75 mm	1.0	0	0
Y	64	0	3.75 mm	0.0	0.997	0.078
Z	35	0	4 mm	0.0	0.078	0.997
T	147	0	3 sec			

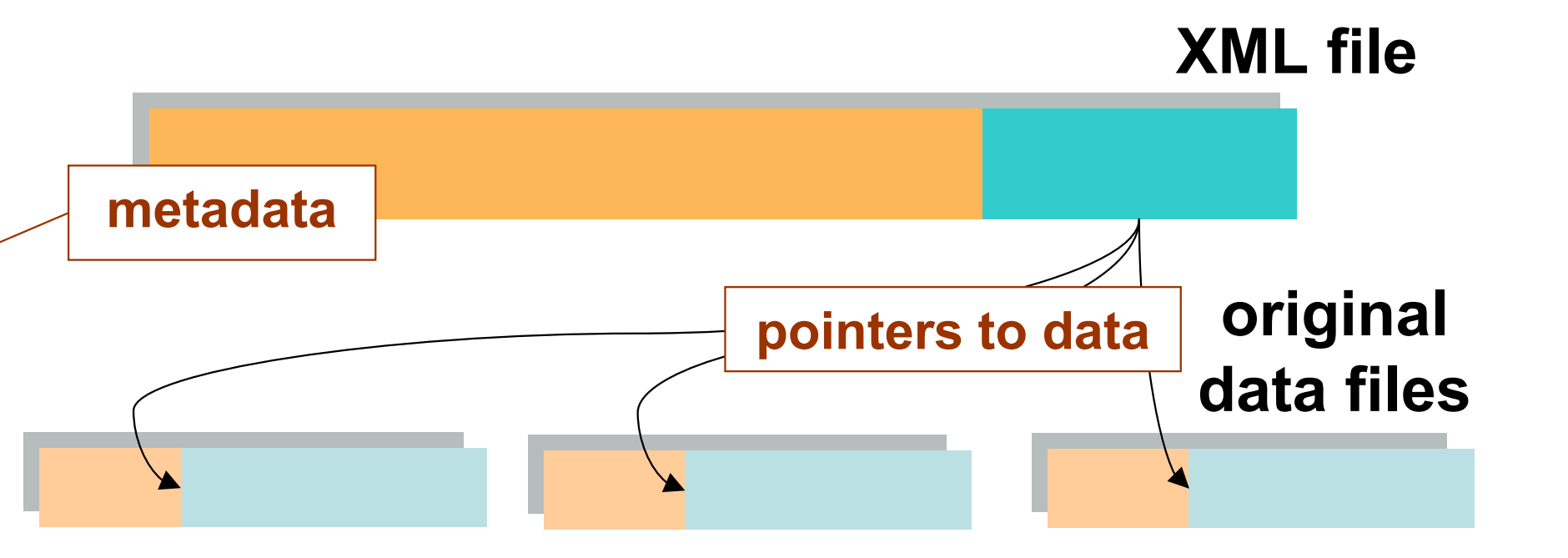

Figure 5: XML files store metadata and pointers to locations of data within the original data files.

4 STATUS

We have created subroutine libraries to provide software support for the XML file format for C, C++, and MATLAB. These libraries employ XML tools based on platform- and language-independent standards such as XPath and DOM. Incorporating these library tools into existing analysis programs allows those programs to read images of any type wrapped by XML files.

We have written tools to create XML files that "wrap" image data in several popular image formats including DICOM, GE Signa 5.x, Analyze 7.5/SPM, NIfTI-1, AFNI BRIK, MINC and others. We have also extended existing visualization and analysis software to read data using the XML files, and also to automatically read metadata (such as orientation and positioning information) that previously required interactive user input.

This general framework, implemented as the BIAC XML Header (BXH), has been in wide use at the Duke-UNC Brain Imaging and Analysis Center (BIAC) since late 2002. This framework has also been adopted as the data interface component of the XML schema developed by the Biomedical Informatics Research Network (BIRN). Both groups receive data from many sources, and greatly benefit from the format-agnostic model of data access.

Extending our raw data with this standard data/metadata interface has enabled us to streamline our data processing and analysis infrastructure by fostering the use of reusable components and has made it straightforward to extend our infrastructure to work properly with new data formats. Retaining the original data in its original form allows us to use a common data interface without requiring data conversion. Finally, by using XML, we provide a human-readable document that can also take advantage of widely-adopted XML standards and software.

ACKNOWLEDGEMENTS & REFERENCES

❖NIH grant IP01NS41328
❖NIH grant 3M01RR00030 (Brain Morphometry BIRN)
❖Duke-UNC Brain Imaging and Analysis Center (BIAC, http://www.biac.duke.edu/)
❖Biomedical Informatics Research Network (BIRN, http://www.nbirn.net/)
❖BIAC XML Header (BXH, http://www.biac.duke.edu/research/bxh/)