

# FSML: Fusion Simulation Markup Language for Interoperability of Data and Analysis Tools \*

Svetlana G. Shasharina and Chuang Li  
Tech-X Corporation, 5621 Arapahoe Ave., Suite A, Boulder, CO 80303  
sveta@txcorp.com and chuli@txcorp.com.

## Abstract

*As the fusion community becomes more interconnected and problems become more complex, very close collaborative efforts are expected. This requires internetworking various codes, comparing solutions from multiple solvers, and sharing of data and data analysis tools. However, the data formats and data analysis tools used in fusion and plasma simulations are highly heterogeneous. Imposing one standard data format and one type of tools is unrealistic due to historical and practical reasons. In this paper, we propose to create the Fusion Simulation Markup Language, or FSML - an XML based system for describing and accessing fusion and plasma physics simulation data of various formats used in the community. The system consists of syntactic and semantic metadata organized in specialized XML schemas and APIs written for accessing data from major data analysis and visualization tools. We present the primary results by wrapping the FSML schemas and APIs in various AVS/Express modules, and by demonstrating its application in two large three-dimension fusion simulation codes M3D and NIMROD. The results show that FSML schema and the set of tools developed will provide a strong initial momentum and technology for the community effort to enhance data exchange and interoperability of analysis tools.*

## 1 Introduction

A central problem for many large-scale scientific and engineering simulations is the processing of various data sets of complex and specialized data storage formats. Many of the formats and tools are developed by teams for internal use and for accommodating users with particular preferences and background. However, modern scientific challenges demand high interoperability between data and var-

ious applications. It is unrealistic to impose one standard storage format. First, it requires a lot of translating of existing data into this new standard format. Second, we cannot anticipate future needs and technology developments. Finally, there are many legacy data sets and programs that are difficult to change. In view of this, many great efforts have been made to structure dataset in a standard form for facilitating the exchange and manipulation of data in recent years.

The Standard Generalized Structured Markup Language (SGML) is one of the first technologies targeting for standardizing and structuring information. Although extremely powerful, SGML has proved to be too complex to be used for general purposes. As a subset of SGML and a specification for designing markup languages, XML was first released in 1998. Because of its extensibility, flexibility, structure and validation, XML is playing an increasingly important role in exchange of a wide variety of data on the World Wide Web ever since [1]. XML defines customized markup languages using Document Type Definition (DTD) and XML Schema. XML Schemas are the successors of DTDs and are richer, more extensible, and more useful. A number of specialized Web-based markup languages have developed based on XML schemas as the standard for information and data exchange on the Web in some specific domains in recent years [2]. These markup languages include Earth Science Markup Language (ESML), Geography Markup Language (GML), Chemical Markup Language (CML), Mathematical Markup Language (MML), Astronomical Markup Language (AML) and others.

In this paper, we propose an XML-based technology, the Fusion Simulation Markup Language (FSML) to enhance the interoperability between different data formats and analysis tools in fusion and plasma simulations community. The FSML technology addresses the heterogeneity of data formats and application tools by employing a common XML schema (FSML schema) to describe the data. The exchange of data in different storage formats among various applications is achieved by describing the data in a FSML instance file and accessing the data through the FSML based parsing

---

\*The work on this project is funded by DOE under contract #DE-FG02-04ER84101 and by Tech-X Corporation

APIs.

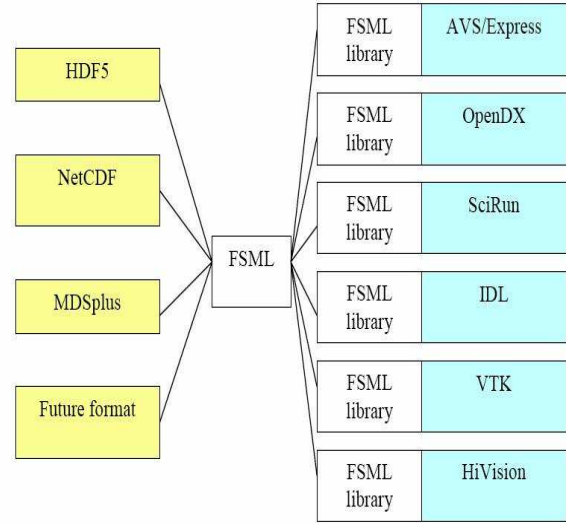
The remainder of this paper is organized as follows. The background of the FSML technology is introduced in Section 2. The design of FSML schema is then reviewed in Section 3. In Section 4, we present the parsing library based on the FSML schema. Section 5 details the application of FSML schema and parsing library in an AVS/Express visualization demonstration. Section 6 provides concluding remarks and prospects of future directions of this project.

## 2 Background

The data formats and data analysis tools used in fusion and plasma simulations are highly heterogeneous. For example, M3D (a three-dimensional Magneto-Hydrodynamics (MHD) code developed at Princeton Plasma Laboratory (PPPL) [3]) and VORPAL (a three-dimensional plasma simulation code developed in University of Colorado and tech-X [4]) use the hierarchical, binary, self-described data format HDF5 [5]. NIMROD, another large three-dimensional MHD multi-organizational code [6], stores most of data in binary Fortran files and MD-Splus trees [7]. It recently started supporting HDF5 as well. Data generated by TRANSP [8], a large transport code developed in PPPL, is stored as MDSplus trees. In fact, the heterogeneity goes deeper. Using the same (HDF5) data format does not provide enough homogeneity. For example, the HDF5 files generated by M3D and NIMROD are organized very differently even for similar simulations. They do not share node structure and do not agree on attributes. They use different names for physically similar variables and store data in different structures.

The data analysis and visualization tools used by different teams are also extremely non-uniform. M3D team developed a set of AVS/Express tools that work well with HDF5 files produced by M3D [9]. The VORPAL team prefers OpenDX and creates a plugin for OpenDX allowing for importing its own HDF5 based output files [10]. The NIMROD data of a particular form of HDF5 can be imported by the SciRun data importer [11], and can also be analyzed by OpenDX and Xdraw. Interactive Data Language (IDL) is heavily used throughout the fusion and plasma simulation community, and it was extended to support the MD-Splus APIs. IDL/MDSplus tools are often preferred to the C APIs. As an example, General Atomics created a rich set of such tools for analyzing simulation and experimental data.

As the heterogeneity of data and application tools increases, the fusion community is becoming more interconnected, and the problems are becoming more complicated. This requires more collaborative efforts, such as comparing and networking of various codes, sharing and exchanging of data between various applications and data analysis tools. For example, the NIMROD collaboration involves partici-

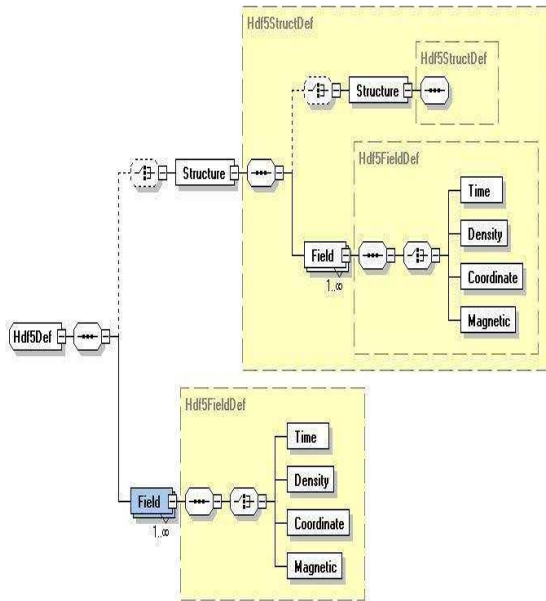


**Figure 1. Providing a middle tier of FSML format and FSML APIs for data analysis tools.**

pants from SAIC, General Atomics in San Diego, the University of Colorado, the University of Wisconsin, and other institutions. The M3D project involves participants from PPPL and the New York University. In both cases, data from a particular simulation might ultimately be compared, analyzed, and visualized at any of the participating institutions. The recent decision of the USA to review its interest in the International Thermonuclear Experimental Reactor (ITER) also requires scientists to face the problem of data and tools unification and interoperability.

The problems outlined above lead to the start of developing the XML-based Fusion Simulation Markup Language. As shown in Figure 1, FSML and its underlying technologies serve as a middle tier between various data formats and analysis tools, and provide a “many-to-one” solution. Applications access data through the FSML-based APIs instead of reading the data directly. Instead of constructing new readers for future data formats for each of the applications, the FSML schema and its parsing APIs will be extended accordingly to provide a uniform solution. The common XML Schema and uniform XML metadata are designed to describe semantically and structurally similar data coming from fusion and plasma simulations. The FSML schema based APIs allow user to parse, query and retrieve data from major data visualization and analysis tools without knowing details of storage formats or converting the original data into native data formats.

The use of XML technology for implementing our solution is based on the following considerations. First, XML is widely supported by industry. There are many useful tools



**Figure 2. Portion of the FSML Semantic Schema (Generated with XMLSpy Schema Editor).**

including parsers, editors, verifiers, and data translators. Second, XML is widely used in many recently developed technologies, which will allow easy integration with them. As an example, Web and Grid services use the XML-based Web Service Definition Language (WSDL) to define the service interfaces. These emerging technologies could provide distribution and mechanisms for interconnected fusion simulations using different models, data formats, and data analysis tools. Third, Common Component Architecture (CCA), a promising solution for multi-component high-performance and distributed applications, supports XML description of components and their interfaces [13]. Using FSML in CCA could facilitate the standardization of components for fusion and plasma physics applications. Finally, metadata has been proved to be very helpful for efficient data archiving and data sharing. One of the successful examples is the Scientific Archive Management system (SAM), which provides mechanisms to easily organize, store, query and retrieve various kinds of the molecular science data. The upcoming Fusion Simulation Project targets to create a flexible framework integrating various fusion models and applications seamlessly [12]. FSML could provide a solution to facilitate the data flow among different elements of simulations using the framework.

### 3 The FSML Schema

The goal of the FSML schema is to design a common set of markups that are suitable for describing the content of MHD computational data. In current work, we use two three-dimensional MHD codes, M3D and NIMROD as foundation to address this effort. Both M3D and NIMROD use a finite element method to solve partial differential equations modeling non-linear resistive MHD. Both codes use HDF5 file to store its data. Based on the outputs of these codes, a prototype of the FSML schema was developed to accommodate the common data structures of these outputs.

The FSML based descriptions are placed in a separate file. Data from different applications and in various formats is described with a unique FSML instance file. The parser reads and interprets the data according to its corresponding FSML description. The challenge here is to create a flexible schema for describing the MHD computational data that can strike a balance between generality and usefulness for this particular domain. A general FSML schema would be easy to extend to describe most common variables in fusion simulation, but difficult to apply in some specific situations. For instance, the physical variables in M3D and NIMROD are mostly the same. However, they are organized fundamentally differently. Some differences are related to design choices, while others are from fundamental differences in the codes themselves. First, NIMROD and M3D are based on different mesh structures. The coordinate data structures in NIMROD are cylindrical (R, Z, and Phi). The (R, Z) plane is discretized using finite-elements and the Phi direction is discretized using Fourier modes. On the other hand, the M3D topology is a series of planes at constant toroidal/cylindrical angle. Each of the planes is discretized with an unstructured mesh of triangles, with data defined at the vertex. The topology of the mesh is constant from plane to plane in M3D. Second, some of the physical variables are fundamentally different. For example, the magnetic field is represented by a three-component vector in NIMROD, but is represented by three scalar potentials in M3D. Third, the way that M3D and NIMROD data are organized in HDF5 is very different. Most nodes values are organized as one or two-dimensional arrays in M3D, but as multi-dimensional arrays in NIMROD output. Finally, M3D and NIMROD often have different names and descriptions for the same physical variables. For example, while NIMROD generally have more straightforward variable names, physical variables in M3D are named as “node\_data[0]”, “node\_data[1]” and so on.

The current FSML prototype schemas provide means to describe the structure and semantic information for selected common variables including magnetic field, temperature field, density fields and geometry information from NIMROD and M3D outputs. The structural or syntactic

metadata describes the structure of the MHD data in terms of bits and bytes. It defines the primitive data types, such as string, integer, float, and double. The syntactic metadata also defines the aggregation of primitive types as compound types including structures and arrays. The current work relies on existing standard formats, such as HDF5, to provide the syntactic metadata when available. It also provides mechanisms to define these metadata for file formats that are not self-describing, such as ASCII and binary formats [2, 14]. The syntactic metadata provides basic information for applications to traverse a data file and retrieve the data. However, an application cannot understand what each piece of data means solely using the primitive and compound type schema definition. The semantic metadata adorn the structural metadata with the added semantics of data in the context of a specific application field (e.g., fusion and plasma simulation). For instance, the semantic metadata tells the parser that a specific array described by the structure metadata is a temperature field or a magnetic field in the computational domain. Examples of the semantic schema definitions in current work include “Magnetic” field, “Temperature” field, “Coordinate” field, and other geometry data (e.g., node connectivity).

Figure 2 shows part of the FSML semantic schema that describes the selected common variables from NIMROD and M3D. As an example, the “Coordinate” element defines the coordinates of mesh nodes in simulation. It also contains attributes that allow users to define further information, such as the mesh structure, units used in discretization, and comments from application developers.

A FSML instance file describing specific fields in output of NIMROD and M3D is illustrated in Figure 3 and Figure 4 respectively. The parser relies on HDF5 tags for detailed syntactic metadata information in these examples, and relies on the semantic tags, e.g., “Temperature”, “Magnetic”, and “Coordinate”, for the specific meaning in MHD simulations. The FSML instance file specifies the path to locate dataset in HDF5 output by assigning appropriate values to various tags such as “Field” and “Structure”. The magnetic field is presented in three separate one-dimensional arrays as three components of a vector in the NIMROD data, and is represented as a single two-dimensional array in the M3D output. The coordinate field is represented in a similar fashion. As described below, in spite of many data structures differences, an unified interface will be provided to access these data for further visualization and analysis.

#### 4 The C++ APIs Parsing FSML-Based Data

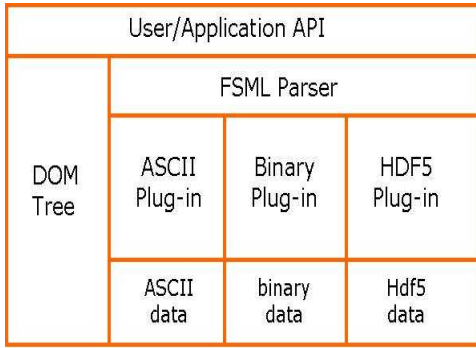
Since C and C++ code can be called from multiple data analysis and visualization tools, we proceeded to design a C++ parsing library based on FSML schema described above. This library provides user APIs for reading FSML

```
<?xml version="1.0"?>
<a:FSML xmlns:a="FSML" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="FSML FSML.xsd">
  <SyntacticMetaData>
    <Hdf5 application="nimrod">
      <Structure name="GRID" instances="1">
        <Field name="X">
          <Coordinate name="x" system="cartesian" mesh="structured"/>
        </Field>
        <Field name="Y">
          <Coordinate name="y"/>
        </Field>
        <Field name="Z">
          <Coordinate name="z"/>
        </Field>
      </Structure>
      <Structure name="step_0001000" instances="1">
        <Structure name="B" instances="1">
          <Field name="X">
            <Magnetic name="x-component" unit="Telsa"/>
          </Field>
          <Field name="Y">
            <Magnetic name="y-component" unit="Telsa"/>
          </Field>
          <Field name="Z">
            <Magnetic name="y-component" unit="Telsa"/>
          </Field>
        </Structure>
      </Structure>
    </Hdf5>
  </SyntacticMetaData>
</a:FSML>
```

Figure 3. A FSML file for NIMROD data.

```
<?xml version="1.0"?>
<a:FSML xmlns:a="FSML" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="FSML FSML.xsd">
  <SyntacticMetaData>
    <Hdf5 application="m3d">
      <Structure name="/" instances="1">
        <Field name="time">
          <Time name="m3d-time" unit="second"/>
        </Field>
      </Structure>
      <Structure name="time_node_data[0]" instances="1">
        <Structure name="node_data[9]" instances="1">
          <Field name="values">
            <Magnetic name="m3d-magnetic" unit="Telsa"/>
          </Field>
        </Structure>
      </Structure>
    </Hdf5>
  </SyntacticMetaData>
</a:FSML>
```

Figure 4. A FSML file for M3D data.



**Figure 5. An overview of architecture of the FSML parsing library.**

definitions, generating the corresponding DOM (Document Object Model) trees, and accessing the data. It uses API plugins for interacting and accessing various types of data formats including ASCII, binary data and HDF5 data. Figure 5 presents an overview of the architecture of FSML parsing library. The library loads in different plugins dynamically based on data formats at runtime, and retrieves data from the original data files according to the FSML description. The DOM trees store the metadata as it is being parsed from the FSML instance file, and allow users to access data set by querying the corresponding metadata. The DOM interface rather than the Simple API for XML (SAX) is used in current implementation due to following reasons. First, the size of current FSML instance file is small. Second, the program needs to access widespread parts of the FSML description for various information. Third, internal data structures of data file are usually complicated. The SAX interface will be considered in our future work for better parsing performance.

Our library was implemented on top of Xerces C++ and ESML. Xerces C++ [15], a freely available versatile XML parsing and processing tool from the Apache XML project, is used to create DOM trees from FSML files. Xerces fully implements the W3C XML, XML Schema, XSLT, Xquery, and XPath recommendations. Some parts of the parsing library referenced the ESML library [14], which is free software (under the terms of the GNU Lesser General Public License published by the Free Software Foundation) from the Information Technology and Systems Center of the University of Alabama.

Our extension to Xerces-C++ and ESML library provides access to special FSML markups, and allows users to access MHD data according to FSML descriptions. To expedite the access to various data, including both the user specified data and simulation data itself, two instances of DOM tree are constructed. One is for the original FSML

description tags, and the other is created upon the first DOM tree with updating based on HDF5 data descriptions.

Currently the parsing library allows users to get fundamental MHD variables by name, location, and time. As in the whole project, our main efforts in the library development concentrate on constructing universal APIs for accessing HDF5 data using different mesh and data structure from various MHD applications, such as M3D and NIMROD. The APIs are then to be used in data analysis and visualization tools agnostic of these differences.

## 5 The Visualization of FSML-based Data

There are many data analysis and visualization tools widely used in the fusion and plasma simulation community, which include IDL (Interactive Data Language), AVS/Express, OpenDX, Xdraw, and SciRun. Both M3D and NIMROD use AVS/Express for visualization. In this section, we demonstrate the application of the FSML parsing library and its underlying FSML schema by constructing various modules in AVS/Express for accessing both M3D and NIMROD simulation results.

AVS/Express is a visual and command-line development tool that enables building reusable objects, application components, and sophisticated data visualization applications. AVS/Express is easily extensible by adding new modules, which are objects with parameters and methods. We extended AVS/Express by adding new modules for accessing and manipulating MHD data from two MHD simulations, M3D and NIMROD. These modules allow users to request and query various MHD outputs through a unified interface.

Our modules are developed using the C++ APIs described in the previous section. To implement this, a static pointer to the “FSMLParser” class is constructed during the initialization of the module. This pointer is then used to access methods defined in the FSML Parsing library.

Several FSML based AVS/Express modules have been implemented for different purposes. Besides the reader module for accessing data, several query modules are developed for making specific manipulation on the input MHD data. Figure 6 shows an example using the FSML based AVS/Express reader to access the MHD data. Users have to provide the location of HDF5 data from a specific MHD simulation and its corresponding FSML descriptions to retrieve data. The outputs of the reader can then be linked to other built-in and user customized modules of AVS/Express for various visualizations. The output ports of the FSML reader module are designed to support most visualizations of general purpose, and include field variables such as magnetic field, temperature field, and other geometry and mesh information. The three-dimensional visualization corresponding to the modules arrangement in Figure 6 is shown in Figure 7. The figure presents the distribution of the mag-

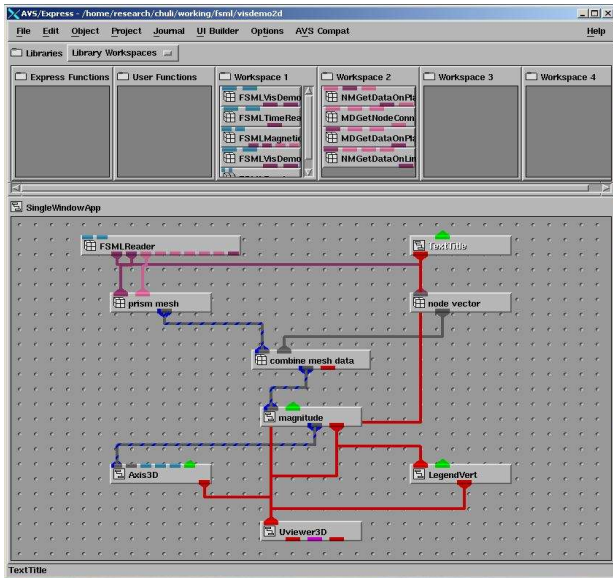


Figure 6. FSML-based AVS/Express Modules.

nitude of magnetic field in an M3D simulation. The exactly same set of modules can be used for the visualization of both M3D and NIMROD HDF5 outputs.

## 6 Conclusions and Future Directions

A prototype of FSML schema has been developed to enhance the interoperability between various data formats and analysis tools used in the fusion and plasma simulation community. The technology under development serves as a middle tier between data and various applications, and thus provides a unified interface to access different data storage formats. The new C++ parsing APIs are constructed based on the FSML schema. Users can access most fundamental field variables from MHD applications through the unified parsing interfaces without knowing details about the data organization. We have demonstrated the application of the FSML based technologies by constructing AVS/Express modules and providing a universal interface for visualizing both NIMROD and M3D data.

The FSML schema will be fully developed for future work. The target data formats will be extended to include not only HDF5, but also ASCII, binary, and the MDSplus format. A full featured C++ parsing library supporting the schema will be provided. More efforts will be made to provide a universal interface for accessing different data and mesh structures. The application of advanced features of XML to support more versatile query and transformation operations on data set is also under consideration.

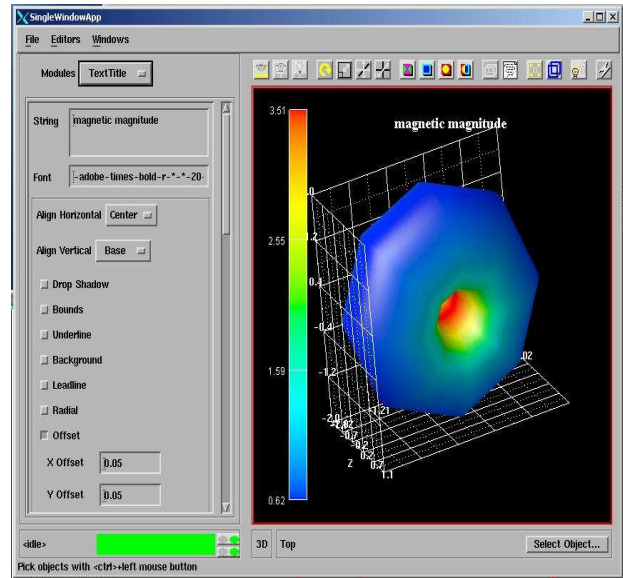


Figure 7. An example of 3-D visualization of the magnitude of magnetic field from M3D.

## 7 Acknowledgment

We thank for Scott Kruger, Scott Klasky, and Joshua Breslau for help with NIMROD and M3D data.

## References

- [1] XML - Extensible Markup Language: <http://www.w3.org/XML> (2005)
- [2] Ramachandran, R., Alshayeb, M., Beaumont, B., Conover, H., Graves, S., Li, X., Movva, S.: Earth science markup language: A solution for generic access to heterogeneous data sets. In: Earth Science Technology Conference. (2001)
- [3] M3D team: <http://w3.pppl.gov/~jchen/> (2004)
- [4] VORPAL team: VORPAL: a Versatile Plasma Simulation Code, <http://www-beams.colorado.edu/vorpal/> (2004)
- [5] NCSA: <http://hdf.ncsa.uiuc.edu/hdf5> (2005)
- [6] NIMROD team: <http://www.nimrodteam.org> (2004)
- [7] MDSplus team: <http://www.mdsplus.org/> (2005)
- [8] Budny, R.V., Ernst, D.R., Hahm, T.S.: Local transport in Joint European Tokamak edge-localized, high confinement mode plasmas with H, D, DT, and T isotopes. *Physics of Plasma* **7** (2000) 5038

- [9] Advanced Visualization Systems:  
<http://www.avs.com/> (2005)
- [10] OpenDX: <http://www.avs.com/> (2005)
- [11] SCIRun:  
<http://software.sci.utah.edu/scirun.html> (2005)
- [12] The FESAC ISOFS Subcommittee: Fusion Simulation Project - Integrated Simulation & Optimization of Fusion Systems (2002)
- [13] CCA: <http://cca-forum.org> (2004)
- [14] ESML - Earth Science Markup Language:  
<http://esml.itsc.uah.edu/index.jsp> (2004)
- [15] Xerces C++ Parser: <http://xml.apache.org/xerces-c/> (2004)