NCML

Proposal of an XML Interface for NetCDF Data

L. Bigagli, S.Nativi

MAY 2000

PIN—University of Florence
Piazza Ciardi, 25, 59100 Prato, Italy
Phone: +39 0574 602535
Fax: +39 0574 602515
Homepage: http://prato2.ing.unifi.it/pin/home.htm

NcML

PROPOSAL OF AN XML INTERFACE FOR NETCDF DATA

INTRODUCTION

The Network Common Data Form (netCDF) is a model for array-oriented data access and storage, designed to be self-describing and network-transparent (i.e. machine-independent). Initially developed by Unidata, it has now been adopted by several organisations as a data access standard. Software bindings of the netCDF interface include C/C++, FORTRAN and Java.[1]

The Extensible Mark-up Language (XML) is a language developed by the W3C, designed to be "straightforwardly usable over the Internet". It supports the definition of customised mark-up languages, by which arbitrary information can be transferred over the Web in a standard manner.[2]

XML is admittedly going to be the Web language in a near future and its support is rapidly growing. This consideration, along with the current shortcomings of transferring large amount of data over the Web, has moved us to consider the encapsulation of netCDF files in XML documents as a suitable solution for meta-data browsing, in the context of Web-shared collections of data.

This document describes the proposed mark-up language, ncML, which was loosely inspired by the work of Mr. Bear Giles[3] and by the Extensible Data Format, developed at the NASA Goddard Astronomical Data Center.[4] This work has been partially funded by the Italian Space Agency (ASI), within project no. RIFR500372 (Sinots) and no. RF374.

Comments can be sent to bigagli@pinet.ing.unifi.it.

[1] See http://www.unidata.ucar.edu/packages/netcdf for more information.

[2] See http://www.w3.org/xml for more information.

[3] See ftp://ftp.unidata.ucar.edu/pub/netcdf/contrib/xml-tools.tar.gz for his "NetCDF ↔ XML toolset".

[4] See http://tarantella.gsfc.nasa.gov/xml for more information.

NCML SPECIFICATION

RATIONALE

The proposed specification observes the following guidelines:

- ASSUMPTION OF NETCDF FILES EXISTENCE — we assume that a ncML document is created from an existing netCDF file, basically to provide the users with a more convenient means of exchanging meta-data information; a ncML document might be created automatically, possibly on-demand from a remote application. Hence, human readability of ncML documents is not important;
- LEXICAL AND STRUCTURAL CONFORMANCE TO CDL— CDL (Common Data Form Language) is the Unidata-proposed notation to provide a textual description of the netCDF data model. Since many tools exist to automate netCDF mapping to and from CDL, we have kept the proposed DTD close to CDL terminology and structure, anticipating future needs of interoperability;
- EXPLOITATION OF XML VALIDITY SEMANTIC — whenever possible, we have mapped netCDF semantic constraints to XML *validity* constraints, to leverage the features of validating XML parsers and ease the design of ncML clients. In particular, mapping netCDF name-uniqueness constraints to XML has raised a non-trivial issue, which is discussed in the next section.

NAME MAPPING

The netCDF interface specifies that dimensions, variables and attributes must be identified by names that be unique in each context. More precisely:

- Distinct dimensions (including the unlimited dimension) must have distinct names;
- Distinct variables must have distinct names;
- Distinct global attributes must have distinct names;
- Distinct attributes associated to the same variable must have distinct names.

Note that this specification allows a variable and a dimension to be identified by the same name (feature exploited in the definition of a *coordinate variable*), as well as attributes associated to distinct variables, etc.

It is convenient to map such names to XML attributes of ID type, since a validating parser would enforce their uniqueness. Unfortunately, in XML, "ID values must uniquely identify the elements which bear them", so the trivial mapping would not cope with such things as coordinate variables and the like.

To solve this problem, according to a common practice, we have defined a name-mangling scheme, by which the context of a netCDF name is encoded with the name itself, providing a unique XML identifier. The resulting name is more obscure to a human reader, but, as stated above in the Rationale section, we are not concerned about ncML readability, since ncML documents are mainly intended to be generated and handled by software components.

The following rules define the name-mangling scheme:

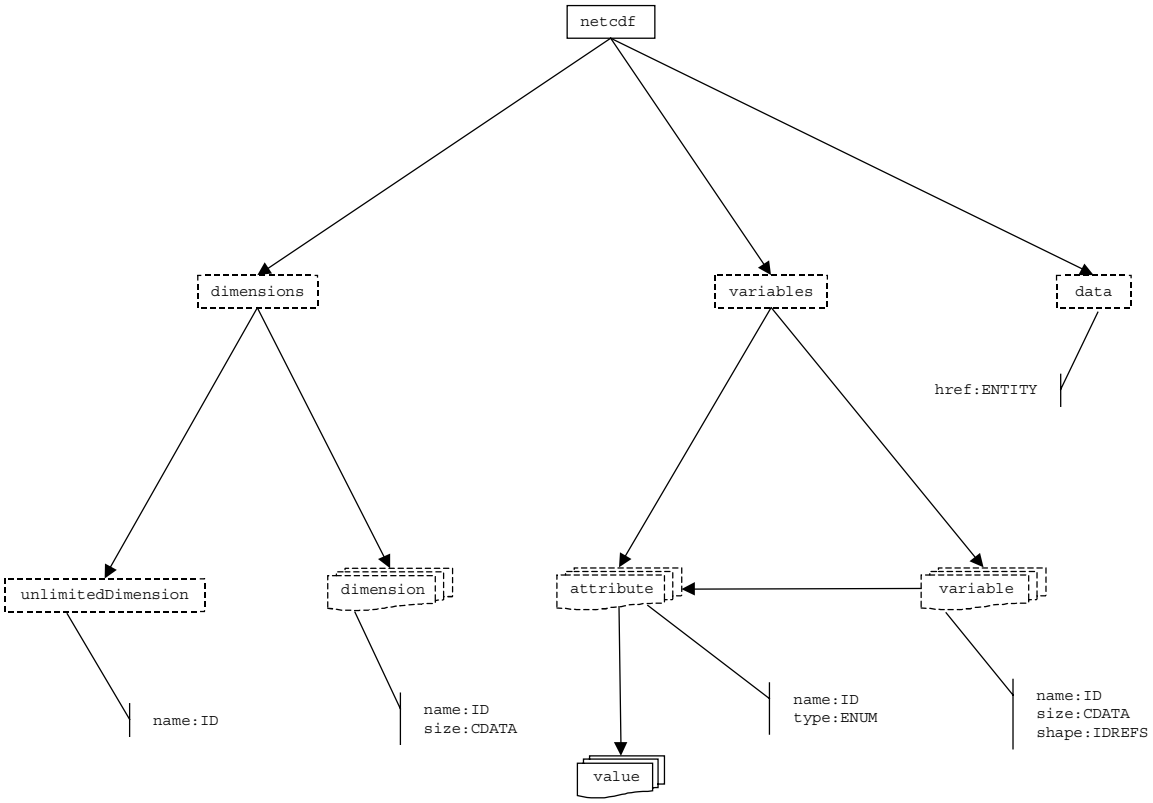
- Dimension (either unlimited or not) names are prefixed by "`d_`"

- Variable names are prefixed by "v_"
- Global attribute names are prefixed by "a_"
- Attribute names associated with variable x are prefixed by "a_x_"

These rules guarantee that a netCDF name in a certain context be mapped to a valid XML ID value on a one-to-one basis. Unfortunately, the above constraints on ncML `name` prefixes can not currently be enforced in XML.[5]

DOCUMENT STRUCTURE

The hierarchical structure of ncML elements, characterised by their main attributes, is depicted in the following scheme.



In the rest of this section, the most relevant parts of the proposed DTD are commented (the whole DTD is listed in the Appendix). DTD fragments commented below may differ from their Appendix counterparts, for the sake of simplicity.

```
<!ELEMENT netcdf (dimensions?,variables?,data?)>
```

Accordingly with the CDL notation, a ncML document "consists of three optional parts", introduced by the mark-up `dimensions`, `variables` and `data`.

[5] XML Schema Specification will enable the expression of such constraints. See http://www.w3.org/XML/Schema.html for more information.

```
<!ELEMENT dimensions (unlimitedDimension?,dimension*)>
```

The `dimensions` part may begin with an `unlimitedDimension`, followed by a list of `dimension` elements. This design enforces the uniqueness constraint on `unlimitedDimension`, as defined by the netCDF interface.

```
<!ELEMENT dimension EMPTY>
<!ATTLIST dimension
    name ID #REQUIRED
    size CDATA #REQUIRED
>
```

As stated in the netCDF specification, "a netCDF dimension has both a name and a size": ncML `dimension` elements contain nothing, but require that attribute `name` and `size` be specified. Attribute `size` must be a positive integer, but currently there is no way to enforce this in XML.[6] Concerning `name` attribute, mapping it to an XML ID type enforces the netCDF uniqueness constraint on dimension names (see the above section, Name Mapping, for further considerations).

```
<!ELEMENT variables (attribute*,variable*)>
```

The `variables` part may contain a list of `attribute`, followed by a list of `variable` elements. The former is the *global attributes* list, in the netCDF terminology.

```
<!ELEMENT attribute (value+)>
<!ATTLIST attribute
    name ID #REQUIRED
    type (byte | char | short | long | float | double | int | real) #REQUIRED
>
```

As stated in the netCDF specification, a netCDF attribute has "a name, a data type, a length, and a value": a ncML `attribute` element contains a list of `value` elements, which implies its length (storing the length in a distinct attribute would undermine the document consistency). The `type` attribute is an XML enumeration of the valid CDL types. Concerning attribute `name`, mapping it to an XML ID type enforces the netCDF uniqueness constraint on attribute names (see the above section, Name Mapping, for further considerations).

```
<!ELEMENT variable (attribute*)>
<!ATTLIST variable
    name ID #REQUIRED
    type (byte | char | short | long | float | double | int | real) #REQUIRED
    shape IDREFS #IMPLIED
>
```

As stated in the netCDF specification, "a variable has a name, a data type, and a shape described by its list of dimensions"; it "may also have associated attributes". Hence, a ncML `variable` element contains a list of `attribute` elements, defined above. The `name` and `type` attributes are identical to those defined in element `attribute`. Attribute `shape` should be a list of `dimension` element names, but currently there is no way to enforce this in XML.[6] Scalar variables have no `shape` attribute.

[6] XML Schema Specification will enable the expression of such constraints. See http://www.w3.org/XML/Schema.html for more information.

```
<!ELEMENT data (#PCDATA)>
<!ATTLIST data
    href ENTITY #IMPLIED
>
<!NOTATION netcdf PUBLIC "-//Unidata//NetCDF 3.4//EN">
```

The `data` part may contain arbitrary text, e.g. an XML `<![ CDATA [ … ]]>` section dumping the associated netCDF file, or its data, or a sampling of the data. We have not investigated this issue, since we consider that the most suitable approach should be just to include a reference to the associated netCDF file, whose existence is assumed in hypothesis (see the Rationale section, above). In fact, dumping large chunks of data would limit the usefulness of encapsulating a netCDF file in a supposedly smaller XML document, in order to allow easier meta-data browsing. Bearing that in mind, we define for the `data` element an attribute `href`, meant to reference an (unparsed) entity: the associated netCDF file. We also define an XML `notation` to be used in the unparsed entity definition.

FURTHER WORK

A few conceivable improvements to the proposed DTD follow:

- Where useful, a more refined structure could be designed for the `data` part (e.g. partitioning data into records, etc.)

- The `entity/notation` mechanism used for referencing the associated netCDF file is somewhat complex and unintuitive, requiring the definition of an external entity and a reference to that in the document; a simpler approach could be to define a MIME data type for netCDF and a declaration like that of the HTML `image` tag for referencing the netCDF file.

APPENDIX

DTD LISTING

The proposed *public identifier* of the DTD, used for referencing it in conforming XML documents, is:

"-//Unidata//NetCDF Mark-up Language 1.0//EN"

A *system identifier* of the DTD that can be used, as well, for referencing it in conforming XML documents, is:

http://oanimal.ing.unifi.it/dtd/netcdf.dtd[L3]

The proposed ncML DTD follows, in its integral form.

```
<?xml version="1.0" encoding="UTF-8" ?>

<!--
 This DTD closely matches the CDL notation: which "consists of three optional
    parts..."

 CDL names are mapped to XML IDs, so we prefix them to enforce uniqueness:
    dimension (either unlimited or not) names are prefixed by "d_"
    variable names are prefixed by "v_"
    global attribute names are prefixed by "a_"
    attribute names associated with variable x are prefixed by "a_x_"
-->

<!NOTATION netcdf PUBLIC "-//Unidata//NetCDF 3.4//EN">
<!ENTITY % CDL_TYPES "(byte | char | short | long | float | double | int |
    real)">

<!ELEMENT netcdf (dimensions?,variables?,data?)>
<!ATTLIST netcdf
 xmlns CDATA #FIXED "http://www.unidata.ucar.edu/netcdf"
>

<!ELEMENT dimensions (unlimitedDimension?,dimension*)>

<!ELEMENT unlimitedDimension EMPTY>
<!ATTLIST unlimitedDimension
 name ID #REQUIRED
>

<!ELEMENT dimension EMPTY>
<!-- "size" attribute must be a positive integer -->
<!ATTLIST dimension
 name ID #REQUIRED
 size CDATA #REQUIRED
>

<!-- "variables and attributes are defined after the 'variables' keyword" -->
<!ELEMENT variables (attribute*,variable*)>

<!ELEMENT attribute (value+)>
<!-- "type" information is implied by the values' syntax in CDL; we make it
    explicit -->
<!ATTLIST attribute
 name ID #REQUIRED
 type %CDL_TYPES; #REQUIRED
```

```
>

<!ELEMENT variable (attribute*)>
<!-- "shape" attribute must be a list of references to dimension IDs -->
<!ATTLIST variable
 name ID #REQUIRED
 type %CDL_TYPES; #REQUIRED
 shape IDREFS #IMPLIED
>

<!ELEMENT value (#PCDATA)>

<!ELEMENT data (#PCDATA)>
<!-- "href" attribute is a link to the associated netCDF file -->
<!ATTLIST data
 href ENTITY #IMPLIED
>
```

A JAVA CONVERTER

We have developed a simple Java converter that generates a conforming ncML document from a given netDCF file.

The converter is packed in an executable .jar file that can be found at:

http://oanimal.ing.unifi.it/dtd/nc2xml.jar

To run it from the command prompt:

- Starting from JRE 1.2, type **`java -jar nc2xml.jar <inputFilePath>[.nc]`**
- With previous JRE, include nc2xml.jar in the classpath and type **`java com.pin.radar.Netcdf2xml <inputFilePath>[.nc]`**

The following optional command-line arguments can be specified:

- **`-o <outputFilePath>`** the path of the generated ncML document. The default is **`<inputFilePath>.xml`**
- **`-base <URLBase>`** the URL base (it must end with a "**/**") of the associated netCDF file, possibly relative to that of the generated ncML document. The default assumes that the netCDF file and the associated ncML document have the same URL base