Report on Time Series Data Service (TSDS) Test Implementation

August 19th, 2012

Authors

B. Cecconi (VOPDC-CDPP/LESIA)
P. Le Sidaner (VOPDC-DIO)
N. Bourrel (CDPP/IRAP)
M. Gangloff (CDPP/IRAP)
B. Renard (CDPP/IRAP)
V. Génot (CDPP/IRAP)
M. Bouchemit (CDPP/IRAP)
E. Budnik (CDPP/IRAP)

Acronyms

AMDA	Automated Multi Dataset Analysis
CDPP	Centre de Données de la Physique des Plasmas
CNES	Centre National d'Etudes Spatiales
DIO	Direction Informatique de l'Observatoire de Paris
HDMC	Heliophysics Data and Model Consortium
IRAP	Institut de Recherche en Astrophysique et Planétologie
LESIA	Laboratoire d'Etudes Spatiales et d'Instrumentation en Astrophysique
OPeNDAP	Open-source Project for a Network Data Access Protocol
TSDS	Times Series Data Service
VOPDC	Virtual Observatory Paris Data Centre

Table of contents

Context

TSDS is a data web-service developed by Bob Weigel (CDS, G. Mason University, USA), Doug Lindholm (LASP, Univ. Colorado, USA), and others. They reported their work at the Heliophysics Data and Model Consortium (HDMC) meeting in Nov. 2011 HDMC meeting, College Park, MD, USA. It is proposed as a standard data service for HDMC datasets.
B. Cecconi, from Observatoire de Paris, participated to that meeting and proposed to test TSDS. He thus applied for a grant from VOPDC to assess TSDS and conduct an implementation test in order to share datasets proposed by the LESIA plasma team, in Meudon (France) and serve them to CDPP, Toulouse (France). We report here on these activities.

Introduction

Time Series Data System (TSDS) is a project that provides a modular Java Servlet-based OPeNDAP server built around the NetCDF-Java implementation of the Unidata Common Data Model and the NetCDF Markup Language for serving time series data. TSDS is serving data from data files on the server side (data provider side) to the user (or the client) with an URL syntax which specifies the desired dataset, the output format as well as some constraints, selection criteria or filtering to be applied to the data.

API

The base-line API builds on OPeNDAP-compliant URL requests of the form:
`http://host/servletpath/dataset.suffix?parameters&constraint&filter`
where

- **`host`**: name of the computer hosting the TSDS servlet;
- *`servletpath`*: is the path to the servlet;
- `dataset`: name of a dataset containing time series parameters;
- `suffix`: type or format of the output;
- `parameters`: list of parameters to return with optional hyperslab (index subset) definitions (default all);
- `constraint`: constraints on the values of the parameters;
- `filter`: filters applied to the parameter values after the constraints have been applied.

The `filter` options include:

- `replace(a,b)` replace any occurrence of the value a with b,
- `replace_missing(a)` replace missing values with the value a,
- `exclude_missing()` exclude any time sample that has a missing value,
- `format_time(format)` format ASCII time output, see Java's SimpleDateFormat[1] (time variable must be explicitly requested to use),
- `stride(n)` return every nth time sample,
- `thin(n)` apply a stride to return about n time samples.

The `constraint` options include:
>, <, >=, <=, and =.

The `suffix` options include:

- `csv`: comma separated values,
- `dat`: tabular ASCII format,
- `bin`: A flat binary table,
- `nc`: Network Common Data Form (NetCDF) file (to be implemented),
- `cdf`: Common Data Format (CDF) file (to be implemented),
- `h5`: Hierarchical Data Format (HDF) version 5 (to be implemented),
- `json`: JavaScript Object Notation (JSON),
- `xml`: An XML representation of the data (to be implemented; structure to be determined),
- `info`: information about the dataset and parameters,
- `html`: HTML view of dataset information and a form for requesting data,
- `dds`: dataset Descriptor Structure (ASCII),
- `das`: dataset Attribute Structure (ASCII),
- `dods`: dataset as defined by the Data Access Protocol (DAP), and
- `asc`: dataset represented as ASCII.

[1] http://download.oracle.com/javase/6/docs/api/java/text/SimpleDateFormat.html

Installation

We followed the TSDS documentation is available on the TSDS website. On that documentation page, we can find an installation and configuration guide. Several installation processes are proposed. We selected that presented in section 3.1 of the TSDS documentation: "Using pre-compiled binary".

It requires to setup a Tomcat server. We selected Tomcat 6.0 downloaded from there:
http://apache.techartifact.com/mirror/tomcat/tomcat-6/v6.0.35/src/apache-tomcat-6.0.35-src.tar.gz
TSDS-20110829 from SourceForge server:
http://sourceforge.net/projects/tsds/files/tsds-20110829.war

The installation is rather easy for users knowing Tomcat. For Tomcat beginners, it is probably not verbose enough. It is necessary to explain quickly where the `.war` file should be installed, how to set up a manager account as well as context files, or at least to provide a link toward a web page where this type of information can be easily found. Once you have installed the `.war` file, the server is working with the examples provided.

Configuration

The data files must be put into the TSDS data directory, located in the `tsds` distribution directory under the name `datasets`. We propose here to move it outside the Tomcat directory. Furthermore, the various datasets should be placed in corresponding sub-direrctories, as explained below. The declaration of the dataset directories is made using a THREDDS catalog. Each dataset is defined in a NcML file.

Moving the TSDS data directory

It is suggested in the documentation that the TSDS files (catalogs, metadata and data) should be placed out of the Tomcat directory, so that everything will not be wiped out during any Tomcat update. The documentation explains how to do that, and mentions that the provided path "*can be a relative or absolute path*". In our testings, only relative paths from the tomcat directory worked. Finally, a tutorial would be welcome to configure the context file.

In our test implementation, we have put the TSDS data directory (here named `tsds_files`) in the same directory as the Tomcat one. See Figure 1. The `web.xml` file must be edited as follows. The following `xml` code:

```
<servlet>
      <servlet-name>TimeSeriesServer</servlet-name>
      <servlet-class>lasp.tss.TimeSeriesServer</servlet-class>
      <init-param>
            <param-name>config</param-name>
            <param-value>tss.properties</param-value>
      </init-param>
      <load-on-startup>1</load-on-startup>
</servlet>
```

shall be modified as:

```
<servlet>
      <servlet-name>TimeSeriesServer</servlet-name>
      <servlet-class>lasp.tss.TimeSeriesServer</servlet-class>
      <init-param>
            <param-name>config</param-name>
            <param-value>../../../tsds_files/tss.properties</param-value>
      </init-param>
      <load-on-startup>1</load-on-startup>
</servlet>
```

The `tss.properties` file must also be modified. The first property (4th line of the file in the version we use) is declaring the location of the dataset directory. Relative paths are not from the `tss.properties` directory, but from the `tsds` one. The default value is:

```
dataset.dir = datasets
```

In our case, it can be replaced by:

```
dataset.dir = ../../../tsds_files/datasets
```

After these changes, we can easily check that the TSDS server works with the provided examples.

THREDDS Catalog files

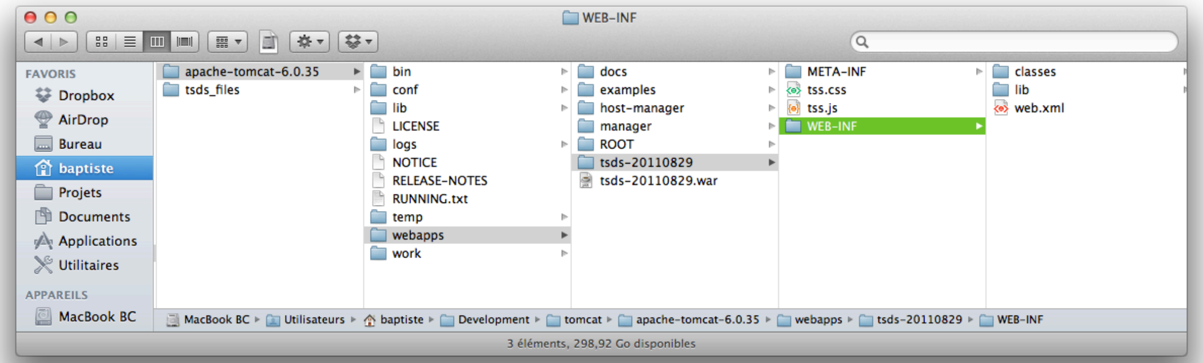
The various data directories are declared in a THREDDS catalog. The provided example is rather elf-explanatory. The main THREDDS catalog located in the `datasets` directory is providing to the system the list of the various datasets and the location of each corresponding THREDDS catalogs.

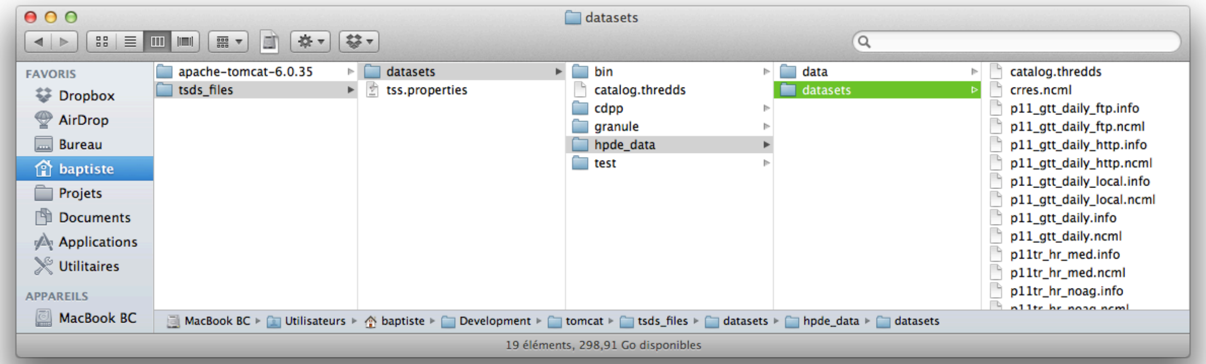
The name of the catalog is set in the main `<catalog>` element, with the `name` attribute. Each dataset THREDDS catalog contains the declaration of two types of services: a `tss` service of type `OpenDAP`; and a `ncml` service of type `NCML`. This declarations shall not be changed:

```
<service name="tss" serviceType="OpenDAP" base="" />
<service name="ncml" serviceType="NCML" base="" />
```

The datasets are declared after the type of services. A typical dataset declaration contains its name, the access information for each service declared at the begining of the file, and some documentation (comments to be displayed by the TSDS server when accesing the dataset information). A simple

Figure 1. *Tomcat directory (top) and `tsds_files` directory (bottom) in our test implementation.*





example is given here (excerpt from the `datasets/test/thredds.catalog` file provided in the distribution):

```
<dataset name="Scalar"
      <access serviceName="tss" urlPath="Scalar" />
      <access serviceName="ncml" urlPath="Scalar.ncml" />
      <documentation type="summary">
            Single variable time series from one binary file
      </documentation>
</dataset>
```

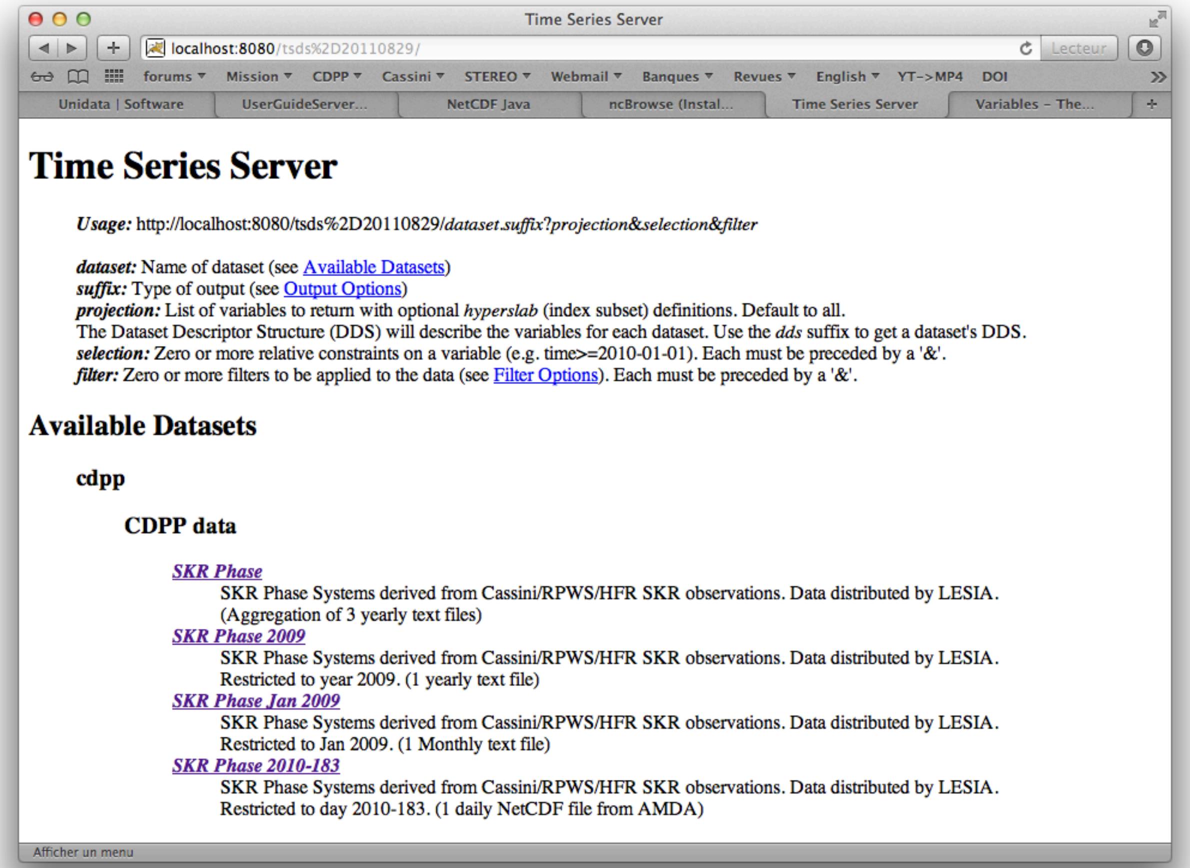
We can put as many datasets as we need into a single THREDDS catalog file. The value of the `urlPath` attribute for the `ncml` service links to the NcML file, which is describing the structure of the data: file (or list of files) and data structure in the data file(s).

The process of creating new THREDDS catalogs is rather easy, and the `README` file of the `datasets/hpde/datasets` directory contains a lot of very useful information in that matter.

NcML Files

The NcML files contain the required information for the TSDS server to access to the data inside the selected dataset: list of associated files, format of the files, and content of the files. An NcML file is an NetCDF XML file that contains: the software interface used to read the data files (this is called IOSP: I/O Service Provider), list of variables (i.e. columns) of the dataset, and list of files if the data is split into several files. The NcML file description will be developed in the next section.

Figure 2. *First page of TSDS server*


Time Series Server
localhost:8080/tsds%2D20110829/
Lecteur
forums Mission CDPP Cassini STEREO Webmail Banques Revues English YT->MP4 DOI
Unidata | Software UserGuideServer... NetCDF Java ncBrowse (Instal... Time Series Server Variables – The...
Afficher un menu

Time Series Server

Usage: http://localhost:8080/tsds%2D20110829/*dataset.suffix?projection&selection&filter*

dataset: Name of dataset (see Available Datasets)
suffix: Type of output (see Output Options)
projection: List of variables to return with optional *hyperslab* (index subset) definitions. Default to all.
The Dataset Descriptor Structure (DDS) will describe the variables for each dataset. Use the *dds* suffix to get a dataset's DDS.
selection: Zero or more relative constraints on a variable (e.g. time>=2010-01-01). Each must be preceded by a '&'.
filter: Zero or more filters to be applied to the data (see Filter Options). Each must be preceded by a '&'.

Available Datasets

cdpp

CDPP data

SKR Phase
SKR Phase Systems derived from Cassini/RPWS/HFR SKR observations. Data distributed by LESIA. (Aggregation of 3 yearly text files)
SKR Phase 2009
SKR Phase Systems derived from Cassini/RPWS/HFR SKR observations. Data distributed by LESIA. Restricted to year 2009. (1 yearly text file)
SKR Phase Jan 2009
SKR Phase Systems derived from Cassini/RPWS/HFR SKR observations. Data distributed by LESIA. Restricted to Jan 2009. (1 Monthly text file)
SKR Phase 2010-183
SKR Phase Systems derived from Cassini/RPWS/HFR SKR observations. Data distributed by LESIA. Restricted to day 2010-183. (1 daily NetCDF file from AMDA)

Testing data server

Interface

Once the THREDDS catalogs are set up, we can test the TSDS server. We disabled the examples provided with the distribution. Hence the first page of the TSDS server is shown on figure 2.

First example: Small text file

As a first example, we tried to share a single file dataset covering 1 month of data with 3 minutes sampling. This file is a text file containing several columns separated by white spaces. The first two columns contain the time in the form: `2010-123 00:30:55`, which is year, day of year, hours, minutes and seconds. The other columns contain floating point values. Here is the beginning of this file:

```
  2008-366 00:00:00 10.809 10.595 186.0 100.9  1788.5 -1778.6
  2008-366 00:03:00 10.809 10.595 187.7 102.6  1788.5 -1778.6
  2008-366 00:06:00 10.809 10.595 189.3 104.3  1788.5 -1778.6
  2008-366 00:09:00 10.809 10.595 191.0 106.0  1788.5 -1778.6
```

The first problem we ran into was that we could not read the date in that form. We had to manually (almost manually...) modify the file before we can make it usable with the TSDS system. We removed the dash and colons of the date, so that the file now looks like:

```
  2008 366 00 00 00 10.809 10.595 186.0 100.9  1788.5 -1778.6
  2008 366 00 03 00 10.809 10.595 187.7 102.6  1788.5 -1778.6
  2008 366 00 06 00 10.809 10.595 189.3 104.3  1788.5 -1778.6
  2008 366 00 09 00 10.809 10.595 191.0 106.0  1788.5 -1778.6
```

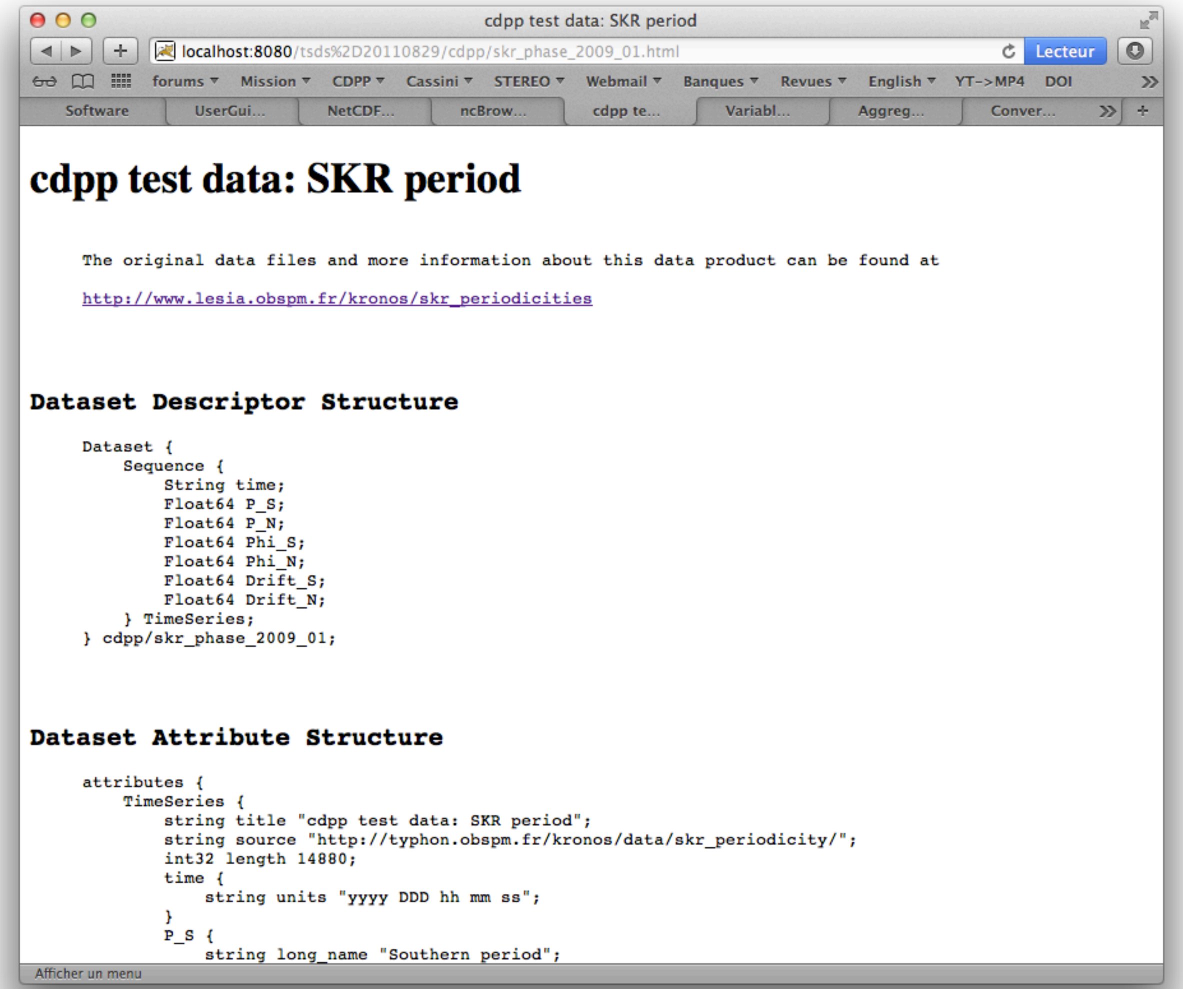
After the reformatting of the date, the data can be served using the following NcML file:

```
<?xml version="1.0" encoding="UTF-8"?>
<netcdf xmlns="http://www.unidata.ucar.edu/namespaces/netcdf/ncml-2.2"
      iosp="lasp.tss.iosp.AsciiGranuleReader"
      location="data/skr_phase/skr_phase_2009_01.txt">

      <attribute name="title" value="cdpp test data: SKR period" />
      <attribute name="source"
            value="http://typhon.obspm.fr/kronos/data/skr_periodicity/"/>

      <dimension name="time" length="14880" isUnlimited="true"/>

      <variable name="time" shape="time" column="1 2 3 4 5" type="String">
            <attribute name="units" value="yyyy DDD hh mm ss"/>
      </variable>

      <variable name="P_S" shape="time" column="6" type="float">
            <attribute name="long_name" value="Southern period"/>
            <attribute name="units" value="h"/>
      </variable>

      <variable name="P_N" shape="time" column="7" type="float">
            <attribute name="long_name" value="Northern period"/>
            <attribute name="units" value="h"/>
      </variable>

      <variable name="Phi_S" shape="time" column="8" type="float">
            <attribute name="long_name" value="Southern Phase"/>
            <attribute name="units" value="deg"/>
      </variable>

      <variable name="Phi_N" shape="time" column="9" type="float">
            <attribute name="long_name" value="Northern Phase"/>
            <attribute name="units" value="deg"/>
      </variable>
      <variable name="Drift_S" shape="time" column="10" type="float">
            <attribute name="long_name" value="SKR peak phase with respect to an
arbitrary period (10.7928h in S)"/>
            <attribute name="units" value="deg"/>
      </variable>

      <variable name="Drift_N" shape="time" column="11" type="float">
            <attribute name="long_name" value="SKR peak phase with respect to an
arbitrary period (10.6h in N)"/>
            <attribute name="units" value="deg"/>
      </variable>

</netcdf>
```

The IOSP we use is `lasp.tss.iosp.AsciiGranuleReader`. The location of the file, from the location of the NcML file is given in the `location` attribute. We first define the dimension of the dataset. In the case of TSDS, only time series my be served, so that the dimension is always `time`. Then we declare the various columns. The `time` variable is treated is a specific way, in order to read the first 5 columns simultaneously. All other columns are declared easily.

Remarks on this example:
- The `column` attribute in the `variable` element is specifically used with the selected IOSP. Indeed, we also tried `lasp.tss.iosp.AsciiIOSP`, but we did not manage to make it work. We noticed in the provided examples that it requires a `orgName` attribute in the `variable` element, but no documentation was found on the attribute.

Figure 3. *TSDS output for `html` info of first example.*


cdpp test data: SKR period
localhost:8080/tsds%2D20110829/cdpp/skr_phase_2009_01.html
Lecteur
forums Mission CDPP Cassini STEREO Webmail Banques Revues English YT->MP4 DOI
Software UserGui... NetCDF... ncBrow... cdpp te... Variabl... Aggreg... Conver...
Afficher un menu

cdpp test data: SKR period

The original data files and more information about this data product can be found at

http://www.lesia.obspm.fr/kronos/skr_periodicities

Dataset Descriptor Structure

```
Dataset {
    Sequence {
        String time;
        Float64 P_S;
        Float64 P_N;
        Float64 Phi_S;
        Float64 Phi_N;
        Float64 Drift_S;
        Float64 Drift_N;
    } TimeSeries;
} cdpp/skr_phase_2009_01;
```

Dataset Attribute Structure

```
attributes {
    TimeSeries {
        string title "cdpp test data: SKR period";
        string source "http://typhon.obspm.fr/kronos/data/skr_periodicity/";
        int32 length 14880;
        time {
            string units "yyyy DDD hh mm ss";
        }
        P_S {
            string long_name "Southern period";
```

- We have declared here one dimension (`time`). If we had dynamic spectral data (variable depending on time and frequency), we would have to define a second dimension. However, it is not clear to us how it would be used here.

Second example: Large text file

On a second example, we take a longer file 1 year of data with 3 minutes sampling. The same NcML file can be used this case apart from changing the file name in the `location` attribute. The selected file is 11MB, and after a few tests of the TSDS server on that dataset, it crashed. The Tomcat logs reported the following error:

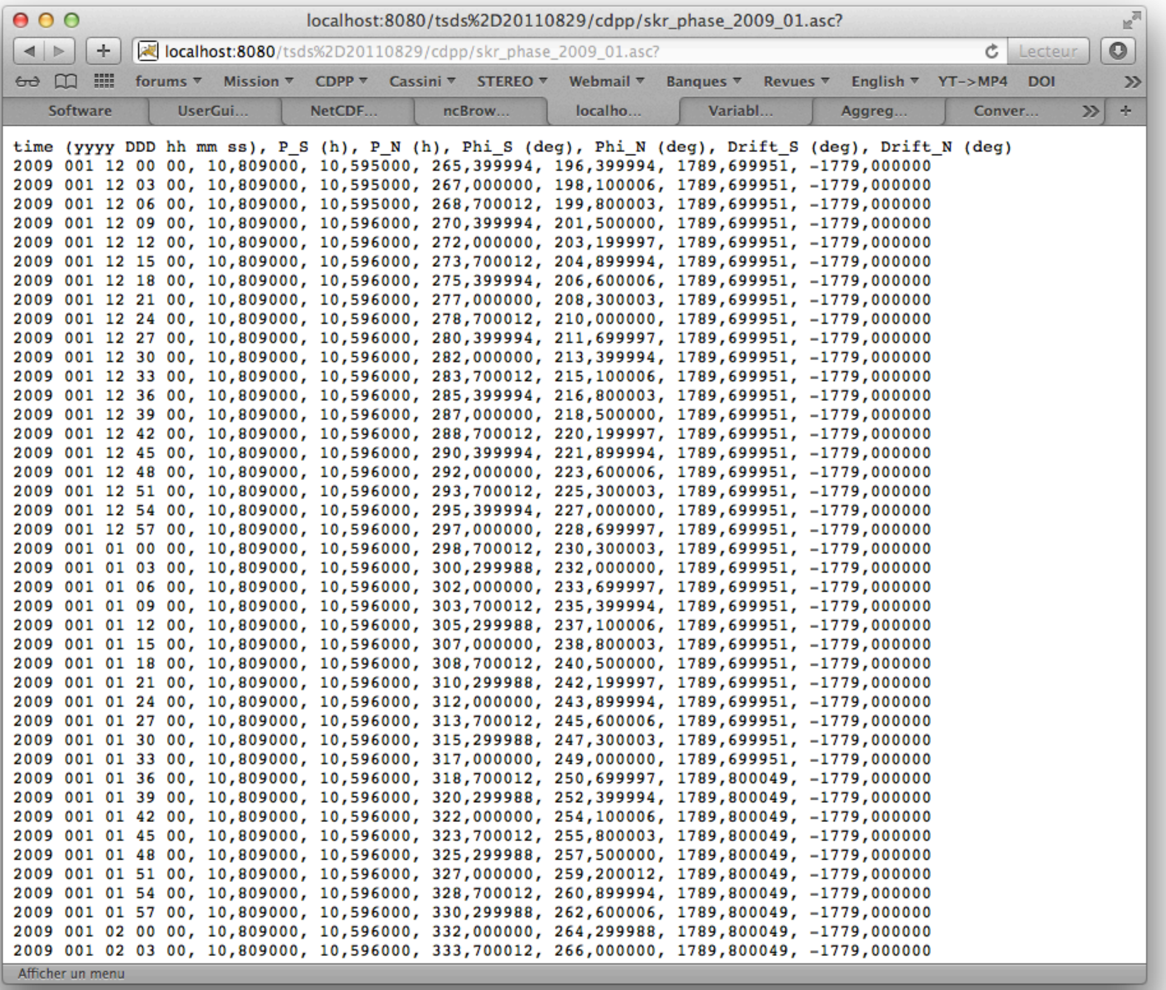
```
Caused by: java.lang.RuntimeException: java.lang.OutOfMemoryError: Java heap space
```

Increasing the memory dedicated to Tomcat solves temporarily (only) the issue. So there may be some memory leak somewhere. Furthermore, serving big ascii files makes TSDS very slow to respond.

Third Example: Series of large text files

Finally, we tried to serve a series of files (same structure as above) but covering several years with the same sampling. In that case, we had to produce a new NcML file using the `aggregation` element available in NcML.

Figure 4. *TSDS output for* `asc` *data of first example.*



```
time (yyyy DDD hh mm ss), P_S (h), P_N (h), Phi_S (deg), Phi_N (deg), Drift_S (deg), Drift_N (deg)
2009 001 12 00 00, 10,809000, 10,595000, 265,399994, 196,399994, 1789,699951, -1779,000000
2009 001 12 03 00, 10,809000, 10,595000, 267,000000, 198,100006, 1789,699951, -1779,000000
2009 001 12 06 00, 10,809000, 10,595000, 268,700012, 199,800003, 1789,699951, -1779,000000
2009 001 12 09 00, 10,809000, 10,596000, 270,399994, 201,500000, 1789,699951, -1779,000000
2009 001 12 12 00, 10,809000, 10,596000, 272,000000, 203,199997, 1789,699951, -1779,000000
2009 001 12 15 00, 10,809000, 10,596000, 273,700012, 204,899994, 1789,699951, -1779,000000
2009 001 12 18 00, 10,809000, 10,596000, 275,399994, 206,600006, 1789,699951, -1779,000000
2009 001 12 21 00, 10,809000, 10,596000, 277,000000, 208,300003, 1789,699951, -1779,000000
2009 001 12 24 00, 10,809000, 10,596000, 278,700012, 210,000000, 1789,699951, -1779,000000
2009 001 12 27 00, 10,809000, 10,596000, 280,399994, 211,699997, 1789,699951, -1779,000000
2009 001 12 30 00, 10,809000, 10,596000, 282,000000, 213,399994, 1789,699951, -1779,000000
2009 001 12 33 00, 10,809000, 10,596000, 283,700012, 215,100006, 1789,699951, -1779,000000
2009 001 12 36 00, 10,809000, 10,596000, 285,399994, 216,800003, 1789,699951, -1779,000000
2009 001 12 39 00, 10,809000, 10,596000, 287,000000, 218,500000, 1789,699951, -1779,000000
2009 001 12 42 00, 10,809000, 10,596000, 288,700012, 220,199997, 1789,699951, -1779,000000
2009 001 12 45 00, 10,809000, 10,596000, 290,399994, 221,899994, 1789,699951, -1779,000000
2009 001 12 48 00, 10,809000, 10,596000, 292,000000, 223,600006, 1789,699951, -1779,000000
2009 001 12 51 00, 10,809000, 10,596000, 293,700012, 225,300003, 1789,699951, -1779,000000
2009 001 12 54 00, 10,809000, 10,596000, 295,399994, 227,000000, 1789,699951, -1779,000000
2009 001 12 57 00, 10,809000, 10,596000, 297,000000, 228,699997, 1789,699951, -1779,000000
2009 001 01 00 00, 10,809000, 10,596000, 298,700012, 230,300003, 1789,699951, -1779,000000
2009 001 01 03 00, 10,809000, 10,596000, 300,299988, 232,000000, 1789,699951, -1779,000000
2009 001 01 06 00, 10,809000, 10,596000, 302,000000, 233,699997, 1789,699951, -1779,000000
2009 001 01 09 00, 10,809000, 10,596000, 303,700012, 235,399994, 1789,699951, -1779,000000
2009 001 01 12 00, 10,809000, 10,596000, 305,299988, 237,100006, 1789,699951, -1779,000000
2009 001 01 15 00, 10,809000, 10,596000, 307,000000, 238,800003, 1789,699951, -1779,000000
2009 001 01 18 00, 10,809000, 10,596000, 308,700012, 240,500000, 1789,699951, -1779,000000
2009 001 01 21 00, 10,809000, 10,596000, 310,299988, 242,199997, 1789,699951, -1779,000000
2009 001 01 24 00, 10,809000, 10,596000, 312,000000, 243,899994, 1789,699951, -1779,000000
2009 001 01 27 00, 10,809000, 10,596000, 313,700012, 245,600006, 1789,699951, -1779,000000
2009 001 01 30 00, 10,809000, 10,596000, 315,299988, 247,300003, 1789,699951, -1779,000000
2009 001 01 33 00, 10,809000, 10,596000, 317,000000, 249,000000, 1789,699951, -1779,000000
2009 001 01 36 00, 10,809000, 10,596000, 318,700012, 250,699997, 1789,800049, -1779,000000
2009 001 01 39 00, 10,809000, 10,596000, 320,299988, 252,399994, 1789,800049, -1779,000000
2009 001 01 42 00, 10,809000, 10,596000, 322,000000, 254,100006, 1789,800049, -1779,000000
2009 001 01 45 00, 10,809000, 10,596000, 323,700012, 255,800003, 1789,800049, -1779,000000
2009 001 01 48 00, 10,809000, 10,596000, 325,299988, 257,500000, 1789,800049, -1779,000000
2009 001 01 51 00, 10,809000, 10,596000, 327,000000, 259,200012, 1789,800049, -1779,000000
2009 001 01 54 00, 10,809000, 10,596000, 328,700012, 260,899994, 1789,800049, -1779,000000
2009 001 01 57 00, 10,809000, 10,596000, 330,299988, 262,600006, 1789,800049, -1779,000000
2009 001 02 00 00, 10,809000, 10,596000, 332,000000, 264,299988, 1789,800049, -1779,000000
2009 001 02 03 00, 10,809000, 10,596000, 333,700012, 266,000000, 1789,800049, -1779,000000
```

```xml
<?xml version="1.0" encoding="UTF-8"?>
<netcdf xmlns="http://www.unidata.ucar.edu/namespaces/netcdf/ncml-2.2">

  <attribute name="title" value="cdpp test data: SKR period" />
  <attribute name="source" value="http://typhon.obspm.fr/kronos/data/skr_periodicity/"/>

  <aggregation dimName="time" type="joinExisting">

    <netcdf iosp="lasp.tss.iosp.AsciiGranuleReader"
            location="data/skr_phase/skr_phase_2007.txt">
      <dimension name="time" isUnlimited="true"/>
      <variable name="time" shape="time" column="1 2 3 4 5" type="String">
        <attribute name="units" value="yyyy DDD hh mm ss"/>
      </variable>
      <variable name="P_S" shape="time" column="6" type="float">
        <attribute name="long_name" value="Southern period"/>
        <attribute name="units" value="h"/>
      </variable>
      <variable name="P_N" shape="time" column="7" type="float">
        <attribute name="long_name" value="Northern period"/>
        <attribute name="units" value="h"/>
      </variable>
      <variable name="Phi_S" shape="time" column="8" type="float">
        <attribute name="long_name" value="Southern Phase"/>
        <attribute name="units" value="deg"/>
      </variable>
      <variable name="Phi_N" shape="time" column="9" type="float">
        <attribute name="long_name" value="Northern Phase"/>
        <attribute name="units" value="deg"/>
      </variable>
      <variable name="Drift_S" shape="time" column="10" type="float">
        <attribute name="long_name"
            value="SKR peak phase with respect to an arbitrary period (10.7928h in S)"/>
        <attribute name="units" value="deg"/>
      </variable>
      <variable name="Drift_N" shape="time" column="11" type="float">
        <attribute name="long_name"
            value="SKR peak phase with respect to an arbitrary period (10.6h in N)"/>
        <attribute name="units" value="deg"/>
      </variable>
    </netcdf>

    <netcdf iosp="lasp.tss.iosp.AsciiGranuleReader"
            location="data/skr_phase/skr_phase_2008.txt">[...]</netcdf>

    <netcdf iosp="lasp.tss.iosp.AsciiGranuleReader"
            location="data/skr_phase/skr_phase_2009.txt">[...]</netcdf>

  </aggregation>
</netcdf>
```

NB: The `[...]` sign stands for the whole data description which is only reproduced in the first `netcdf` element.

As it can be seen, the aggregation of files is easy to describe. However, the description of the data structure must be repeated for each `netcdf` file description.

Serving these files (3 x 10MB) works only with a large increase of the memory dedicated to Tomcat. Furthermore, after a few requests, the server fails with a `Java heap space` error requiring to restart Tomcat.

Fourth Example: NetCDF file

We tried serving from a NetCDF file containing the same data set, but we did not succeed. The tested NetCDF file is currently in use in the AMDA tool developed by CDPP in Toulouse, so it is functional. The result of the `ncdump` command on the file is provided below:

```
netcdf /Users/baptiste/Development/tomcat/tsds_files/datasets/cdpp/data/cdpp/data/
testng.nc {
 dimensions:
   Time = UNLIMITED;   // (4322 currently)
   TimeLength = 17;
   Dim_1 = 1;
 variables:
   char Time(Time=4322, TimeLength=17);
   float P_S(Time=4322);
   float P_N(Time=4322);
   float Phi_S(Time=4322);
   float Phi_N(Time=4322);
   float Drift_S(Time=4322);
   float Drift_N(Time=4322);
   char StartTime(TimeLength=17);
   char StopTime(TimeLength=17);
}
```

From this, we have built an NcML file to serve the NetCDF file, which you can find here:

```
<?xml version="1.0" encoding="UTF-8"?>
<netcdf xmlns="http://www.unidata.ucar.edu/namespaces/netcdf/ncml-2.2"
  location="data/skr_phase_nc/skrp_2010183000000000.nc">

  <attribute name="title" value="cdpp test data: SKR period" />
  <attribute name="source" value="http://typhon.obspm.fr/kronos/data/skr_periodicity/"/>
  <dimension name="Time" isUnlimited="true"/>
  <dimension name="TimeLength" length="17"/>
  <variable name="Time" shape="Time TimeLength" type="char">
    <attribute name="units" value="ISO-8901" />
  </variable>
  <variable name="P_S" shape="Time" type="float"/>
  <variable name="P_N" shape="Time" type="float"/>
  <variable name="Phi_S" shape="Time" type="float"/>
  <variable name="Phi_N" shape="Time" type="float"/>
  <variable name="Drift_S" shape="Time" type="float"/>
  <variable name="Drift_N" shape="Time" type="float"/>
  <variable name="StartTime" shape="TimeLength" type="char">
    <attribute name="units" value="ISO-8901" />
  </variable>
  <variable name="StopTime" shape="TimeLength" type="char">
    <attribute name="units" value="ISO-8901" />
  </variable>
</netcdf>
```

Although you can get to the dataset webpage, the type of the Time is not recognized (`Float64` displayed instead of `Char`). With the expected exception of the last two variables, all other variables are correctly recognized and processed by TSDS.

The variable `Time` in the NetCDF file is a 17 character string with the following coding: `yyyyDDDhhmmssSSS` (according to Java SimpleDateFormat class definition). We tried to replace the time variable description by

```
  <variable name="Time" shape="Time" type="string">
    <attribute name="units" value="yyyyDDDhhmmssSSS" />
  </variable>
```

with no success. The TSDS server fails with the following message:

```
Unable to write the response.
```

Summary and Open points

- TSDS runs over Tomcat. Installation is straightforward.
- Repository description (listing of datasets) is done using THREDDS catalogs. They are easy to write.
- Dataset description is done through NcML files. NcML files appeared to be more difficult to create than catalog files. There is a clear need for a description of NcML files, of their content and of their use.
- On the server side, access to files is done through IOSPs (software interfaces). A list of existing IOSPs would be very welcome, as well as their specific options and attributes to be used when declaring the variables.
- Need for a NCML file validator: in terms of syntax and in terms of adequation with data file.
- Need for detailed help/guidelines for each IOSP (it seems that NcML syntax differs from one IOSP to the other)
- With this architecture, we suspect that all files of a dataset should have the same data structure.
- How do we take care of comment lines in text data files ?
- Date formatting syntax in data file can lead to problems (eg: 2004-300 00:10:00 in text files, or strings in NetCDF files).
- Add VOTables output ? In this case, we must include an additional XML file for metadata per dataset on server side to construct proper VOTable headers.

Conclusion

TSDS is a very promising way to serve data. The model used is very simple and compelling: data are described in a simple framework and the server infrastructure is ready to use. On the client side, we just have to form a request URL including filters to get the data in the desired format. Furthermore, it is using OPeNDAP, which is a standard. Our implementation test shows it is working well on simple time series data, with a restriction on the dataset size.

Finally, we feel that the following aspects should be improved: (1) More guidelines and step by step tutorials are required for data providers; (2) The implementation done using Tomcat is too slow to be usable on large datasets.

Online Resources

OPeNDAP: http://www.opendap.org
TSDS: http://tsds.net/ or http://timeseries.org/
documentation: http://tsds.net/doc
SourceForge project: http://sourceforge.net/projects/tsds/
NcML: http://www.unidata.ucar.edu/software/netcdf/ncml/
THREDDS: http://www.unidata.ucar.edu/projects/THREDDS/tech/catalog/v1.0.2/InvCatalogSpec.html
NetCDF-java: http://www.unidata.ucar.edu/software/netcdf-java/tutorial/
(details on NcML and IOSP can be found there)