Building the resource descriptor for your EPN-TAP service in DaCHS

Work in Progress

Please don't use this Document for now, we are currently updating it.

EPN2020-RI

EUROPLANET2020 Research Infrastructure

H2020-INFRAIA-2014-2015

Grant agreement no: 654208

Document: VESPA-WP6-3-020-TP-v0.1(53)

Building the resource descriptor for your EPN-TAP service in DaCHS

Date: 2021-03-31

Start date of project: 01 September 2015 Duration: 48 Months

Responsible WP Leader: Stéphane Erard

Project co-funded by the *European Union's Horizon 2020 research and innovation programme*		
Dissemination level		
PU	Public	☐
PP	Restricted to other programme participants (including the Commission Service)	☐
RE	Restricted to a group specified by the consortium (including the Commission Services)	☐
CO	Confidential, only for members of the consortium (excluding the Commission Services)	☐

Project Number	654208
Project Title	EPN2020 - RI
Project Duration	48 months: 01 September 2015 – 30 August 2019

Document Number	WP6-task3-020-v0.1
Delivery date	30 Jun 2017
Title of Document	Building the resource descriptor for your EPN-TAP service in DaCHS
Contributing Work package (s)	WP6
Dissemination level	PU
Author (s)	Baptiste Cecconi

Abstract:

Document history (to be deleted before submission to Commission)

Date	Version	Editor	Change	Status
06 Apr 2017	0.0	Baptiste Cecconi	Initial version	DRAFT
10 Apr 2017	0.1	Baptiste Cecconi	Added sections "Service Metadata" and "EPNcore Table Definition"	DRAFT

Table of Contents

- Introduction
- Overall Structure
- Service Metadata
- EPNcore table definition
- Data ingestion

Introduction

This document describes how to build your Resource Descriptor (RD) for an EPN-TAP service using DaCHS. The full documentation of DaCHS is available at http://docs.g-vo.org/DaCHS/ref.html, including a section "Anatomy of an RD" that describes the RD structure and syntax in details.

Overall Structure

The RD file is an XML file. The properties of the RD file can be either set up as XML children elements or as attributes of their parent property. The two following examples are equivalent, the first show the attribute syntax, while the second illustrates the XML element child option.

- Attribute syntax:

```
<resource schema="my_service">
        [...]
</resource>
```

- Child syntax

```
<resource>
        <schema>my_service</schema>
        [...]
</resource>
```

The latter syntax is useful when the property has children properties.

Service Metadata

The first property of an RD is the list of service metadata. They are specified in a series of `<meta>[...]</meta>` elements. The following `<meta>` elements should be present in your file:

<meta> element name attribute	**Content**	**Example**
title	The title of your resource. This is the title of your database. This should be rather explicit, basically, the meaning of the acronym or the short description of the service.	`<meta name=" title">Nancay Decameter Array observation database</meta>`

description	The description of you resource. This is long description. Put here anything that could be useful to understand the content or find the resource with full text search engines. Place yourself in the skin of your fellow scientists when writing this part. This must be understandable by non-specialist scientists	<meta name="description" format="plain"> Decametric radio observation from Nancay decameter array. The Nancay Decameter Array (NDA) at the Station de Radioastronomie de Nancay (SRN) is a phased array of 144 "Teepee" helicoidal antenna, half of which being Right Handed (RH) polarized and the other half being Left Handed (LH) polarized. Four receivers are currently connected to the NDA, sampling data in spectral ranges within 5 to 80 MHz. </meta>
copyright	This contains the copyright, rules of use and acknowledgments related to the resource and the data served by the resource. Indicate here the distribution licence if there is one selected. Specify the "rules of use" or "rules of the road", or "data use policy"... You can also give acknowledgment policy and citation rules.	<meta name="copyright"> Rules of Use: SRN/NDA observations in open access can be freely used for scientific purposes. Their acquisition, processing and distribution is ensured by the SRN/NDA team, which can be contacted for any questions and /or collaborative purposes. Contact email: contact.nda@obs-nancay.fr We kindly request the authors of any

communications and

publications using these data to let us know about them,
include minimal citation to the reference and

acknowledgements as presented below.

Acknowledgement:
The authors acknowledge the Station de Radioastronomie de
Nancay of the Observatoire de Paris (USR 704-CNRS, supported
by University d'Orleans, OSUC, and Region Centre in France)
for providing access to NDA observations accessible
online at http://www.obs-nancay.fr

Reference:
A. Lecacheux, The Nancay Decameter Array: A Useful Step
Towards Giant, New Generation Radio Telescopes for Long

Wavelength Radio Astronomy, in Radio Astronomy at Long

Wavelengths, eds. R. G. Stone, K. W. Weiler, M. L. Goldstein,
and J.-L. Bougeret, AGU Geophys. Monogr. Ser., 119, 321, 2000.
</meta>

creationDate	The creation date of the resource descriptor (ISO-8601 formatted)	`<meta name="creationDate">2016-03-30T15:52:00</meta>`
creator.name	The name of the creator of the resource (can be a person or an institute)	`<meta name="creator.name">Station de Radioastronmie de Nancay</meta>`

subject	There can be as many `<meta name="subject">` metadata elements as needed. The values to input here should be taken from the IVOA flavored Unified Astronomy Thesaurus (UAT) when available.
	
	At least one of the top-level keywords of the UAT must be provided (See this page). The typical list of interest for VESPA is:
	
	▪ exoplanet-astronomy
	▪ solar-physics
	▪ solar-system-astronomy
	▪ planetary-science
		`<meta name="subject">jupiter</meta>`
		`<meta name="subject">the-sun</meta>`
		`<meta name="subject">solar-radio-emission</meta>`
		`<meta name="subject">aurorae</meta>`
		`<meta name="subject">planetary-magnetosphere</meta>`
		`<meta name="subject">solar-wind</meta>`
		`<meta name="subject">radio-astronomy</meta>`
		`<meta name="subject">solar-system-astronomy</meta>`

contact.email	The email address for questions and requests about the service. It is preferable to provide the users with an alias email that points to a one or few persons in your team. Having a real person email here may break the process if that person leaves your institute and you don't update the resource descriptor.	`<meta name="contact.name">Laurent Lamy</meta>`
contact.name	The name of the person to contact for questions and requests about the service	`<meta name="contact.email">contact.nda@obs-nancay.fr</meta>`

contact.address	The real mail address of the institution or data center that distributes the resource.	`<meta name="contact.address">Station de Radioastronomie Route de Souesmes, F-18330 Nancay, France</meta>`
referenceURL	An http URL that points to a description of the resource	`<meta name="referenceURL">https://www.obs-nancay.fr/reseau-decametrique/</meta>`
facility	If you are serving observational data, you can give here the name of the observatory / spacecraft. Note that several names (including acronyms) could be provided in a #-separated list (see example).	`<meta name="facility">Station de Radioastronomie de Nancay#SRN</meta>`
instrument	If you are serving observational data, you can give here the name of the telescope / experiment / instrument.	`<meta name="instrument">Nancay Decameter Array#NDA</meta>`
source	This should be an ADS bibcode to a paper presenting the resource of the data present in the resource.	`<meta name="source">2000GMS...119..321L</meta>`

ContentLevel	In general, there are 4 elements of those, with the following values: "General", "University", "Research", "Amateur". You can restrict the list.	`<meta name="contentLevel">General</meta>`
		`<meta name="contentLevel">University</meta>`
		`<meta name="contentLevel">Research</meta>`
		`<meta name="contentLevel">Amateur</meta>`

EPNcore table definition

The EPNcore table should be defined in the RD using the `epntap2` mixin. This ensures that your EPN-TAP service is compliant with the EPNcore specification. The `epntap2` mixin will be updated as needed via the DaCHS debian package update. If you only plan to use the EPNcore mandatory parameters, your table definition section will be very simple:

```
<table id="epn_core" onDisk="true" adql="True" primary="granule_uid">
        <mixin spatial_frame_type="body">//epntap2#table-2_0</mixin>
</table>
```

This minimal table definition says:

- define an `epn_core` table
- write it on disk (i.e., do not keep it in RAM)
- activate ADQL for query
- use the `granule_uid` column for the primary key of the table
- use the `epntap2` template table with only mandatory parameters, and with `spatial_frame_type = "body"`

The `spatial_frame_type = "body"` attribute is required, even if you don't use the spatial coordinate columns, as the mixin has to know what to put into the column headers. The spatial coordinate columns' definitions depend on the `spatial_frame_type`.

If you plan to use some optional parameters, as defined in the EPNcore specification, the table definition will look like:

```
<table id="epn_core" onDisk="true" adql="True" primary="granule_uid">
        <mixin
                spatial_frame_type="body"
                optional_columns="access_url access_format access_estsize thumbnail_url
publisher bib_reference target_region feature_name"
                >//epntap2#table-2_0</mixin>
</table>
```

The extra `optional_columns` attribute tells the template engine to set up those extra columns, as they are defined in the `epntap2` mixin.

If you plan to use custom columns of your own, you have to define them in the table definition element, as shown in the following example:

```
<table id="epn_core" onDisk="true" adql="True" primary="granule_uid">
        <mixin
                spatial_frame_type="body"
                optional_columns="access_url access_format access_estsize thumbnail_url
publisher bib_reference target_region feature_name"
                >//epntap2#table-2_0</mixin>
        <column name="receiver_name" type="text" ucd="meta.id" description="Receiver name
used with the instrument." />
</table>
```

The `column` elements defines an extra column of the EPNcore table.

Data ingestion

The metadata ingestion is done through module called a *Grammar* in DaCHS jargon. Depending on the form of the metadata, different solutions are available. The Grammar module output is fed to a *rowmaker* module, which fills the table rows, with transformations if necessary.

- *Preprocessed metadata available as a CSV file:* The data provider is pre-processing his data collection to build a CSV file, containing the EPNcore metadata using the adequate units and conventions. In this case, the *csvGrammar* shall be used.
- *Individual data files available from the DaCHS server as FITS files:* The data provider is mounting a remote volume (e. g., through NFS) with the data files. If the data format is FITS, we can use the *fitsProdGrammar* to load the FITS files header.
- *Individual data files available from the DaCHS server as CDF files:* The data provider is mounting a remote volume (e. g., through NFS) with the data files. If the data format is CDF, we can use *cdfHeaderGrammar* to load the CDF global attributes.
- *The metadata is available in an external SQL database:* The data provider has access to an SQL database, containing the metadata (or data) he wants to load into his service. In this case, the odbcGrammar shall be used.
- *If all previous cases don't apply:* The data provider should use a *customGrammar* to load the metadata into DaCHS, through a dedicated python script.

We show below a simple example with CSV files available from the resource descriptor directory.

```
<data id="import">
        <!--        Define where to retrieve the data        -->
        <sources>
                <!-- Pattern is used when there are multiple source files -->
                <!-- (here all the .csv files, and a data directory next to the q.rd file)
-->
                <pattern>data/*.csv</pattern>
        </sources>

        <!-- we use the csvGrammar on the files defined in sources -->
        <csvGrammar/>

        <!-- now we send the data to the epn_core table -->
        <make table="epn_core">

                <!-- Inserts the data of each row made by the grammar in its column
-->
                <rowmaker idmaps="*">
                <!-- idmaps="*" implies that any CSV columns with the same name as the
epn_core column is mapped without processing -->

                        <!--        Insert non-varying data        -->
                        <var key="target_name">"Mars"</var>
                        <var key="service_title">"\schema"</var>
                        [...]
                        <!-- Bind the columns required by EPN-TAP        -->
                        <apply procDef="//epntap2#populate-2_0" name="fillepn">
                                <bind key="granule_uid">@granule_uid</bind>
                                <bind key="granule_gid">@granule_gid</bind>
                                <bind key="obs_id">@obs_id</bind>
                                [...]
                        </apply>
                </rowmaker>
        </make>
</data>
```

We list below a series of repositories using various grammar types:

Grammar type	Provider	Service name	Repository URL
customGrammar	PADC	bass2000	https://voparis-gitlab.obspm.fr/vespa/dachs/services/padc/voparis-tap-helio/bass2000
	PADC /MASER	voyager_pra	https://gitlab.obspm.fr/maser/voparis-tap-maser/voyager_pra
	PADC /MASER	expres	https://gitlab.obspm.fr/maser/voparis-tap-maser/expres
	PADC/CDN	nda	https://gitlab.obspm.fr/maser/vogate-obs-nancay/nda
	IDOC	gaia_dem	https://voparis-gitlab.obspm.fr/vespa/dachs/services/idoc/idoc-dachs.ias.u-psud.fr/medoc/gaia_dem
	FHNW.CH	ecallisto	https://voparis-gitlab.obspm.fr/vespa/dachs/services/fhnw.ch/tap.cs.technik.fhnw.ch/ecallisto
	FHNW.CH	rhessi_flares	https://voparis-gitlab.obspm.fr/vespa/dachs/services/fhnw.ch/tap.cs.technik.fhnw.ch/rhessi_flares
csvGrammar	Jacobs	mars_craters_lagain	https://voparis-gitlab.obspm.fr/vespa/dachs/services/jacobsuni/mars_craters_lagain
	Jacobs	mars_craters	https://voparis-gitlab.obspm.fr/vespa/dachs/services/jacobsuni/mars_craters
	Jacobs	planmap	https://voparis-gitlab.obspm.fr/vespa/dachs/services/jacobsuni/planmap
	Jacobs	planetserver_crism	https://voparis-gitlab.obspm.fr/vespa/dachs/services/jacobsuni/planetserver_crism
	Jacobs	usgs_wms	https://voparis-gitlab.obspm.fr/vespa/dachs/services/jacobsuni/usgs_wms
	Jacobs	planetserver_m3	https://voparis-gitlab.obspm.fr/vespa/dachs/services/jacobsuni/planetserver_m3
	FHNW.CH	iris_obs	https://voparis-gitlab.obspm.fr/vespa/dachs/services/fhnw.ch/tap.cs.technik.fhnw.ch/iris_obs
odbcGrammar	IDOC	eit_syn	https://voparis-gitlab.obspm.fr/vespa/dachs/services/idoc/idoc-dachs.ias.u-psud.fr/medoc/eit_syn