International
Virtual
Observatory
Alliance

Observation Data Model Core Components
Version 1.1

IVOA Working Draft December 2, 2010

This version:
http://www.ivoa.net/Documents/Notes/WD-ObsCoreDM-1.1-20101202.pdf

Latest version:
http://www.ivoa.net/Documents/Notes/WD-ObsCoreDM-1.0-20100701.pdf

Previous versions:
1.0

Editor(s):
Mireille Louys, François Bonnarel, Patrick Dowler, Alberto Micol

Authors:
Mireille Louys, François Bonnarel, Alberto Micol,
David Schade, Anita Richards, Patrick Dowler,
Laurent Michel, Jesus Salgado, Doug Tody,
Igor Chilingarian, Daniel Durand, Bruno Rino

Abstract

This document discusses the definition of the **core components** of the Observation data model that are necessary to perform data discovery when querying data centers for observations of interest. It exposes the use-cases to be carried out, explains the model and provides guide lines for its implementation as a TAP service based on the Table Access Protocol (TAP). Such a simple model is easy to understand and to implement by data providers

that wish to publish their data into the Virtual Observatory. This effort which integrates data modeling and data access aspects in a single service is named Obs/TAP and will be referenced as such in the IVOA registries. This is not covering the full Observation data model that will be developed in another IVOA document, and will include a description of the Provenance of an Observation.

Status of this document

This document has been produced by the Data Model Working Group in coordination with partners from DAL and VOQL WG. On one hand it describes the metadata attached to an astronomical observation and develops an abstract view of the Observation data model Core Components part. On the other hand it contains a guide for implementing in the Table Access Protocol framework and covers the data access layer aspects of this effort.

Due to the DM and DAL aspects of this document, this will circulate and be reviewed by both Working Groups.

Acknowledgements

This work has been partly funded by Euro-VO AIDA project that we acknowledge here. SSC XMM Catalog service supported the implementation of the SAADA version of Obs/TAP at Strasbourg Observatory. The Canfar (tb completed , corrected ???) project also contributed for the implementation of Obs/TAP at CADC, Victoria.

Contents

1 Introduction

Modeling of observational metadata has been an important activity of the IVOA since its creation in 2002. Various modeling efforts like the Resource Metadata, the Space Time Coordinates (STC), the Spectrum, and the Characterisation data models have become approved IVOA standards and are currently used in IVOA services and applications.

Meanwhile, the Table Access Protocol (TAP) was developed and reached recommendation status in March 2010. TAP defines a service protocol for accessing tabular data, like astronomical catalogues, or, more generally, database tables. TAP basic idea is to allow a client to (step 1) browse through the various tables and columns (names, units, etc.) to collect the information necessary to then (step 2) actually perform a tabular query.

Building on the two above-mentioned important blocks, data providers have now the ability to expose data model-based information through the TAP protocol.

The third fundamental building block relevant to this effort is an initiative of the IVOA Take Up Committee that, in the course of 2009, collected a number of use cases for data discovery (see Appendix A). Those use cases aim at providing an astronomer the possibility to pose a world-wide query for scientific data of certain characteristics and to eventually retrieve the relevant data products. The ability of posing scientific queries to several data centres at once is a fundamental case for the Virtual Observatory.

The last step for an homogeneous access across all data centres, which would allow a client to send one and the same query to all TAP services that implement the same model, is the stipulation that a data model is exposed (through TAP) using agreed naming conventions, formats, units, and reference systems. This is what this document is for.

The purpose of this document is twofold: (1) to define a simple core data model for observational metadata, and (2) to define the standard way to expose it through the TAP protocol, so to be able to provide a uniform interface to observational data products. The herein described ObsTAP protocol is the answer to the questions posed by the Take Up Committee (2009) now changed to Standing Committee for Science Priorities(May 2010).

The document is composed as following:

- Section 2 briefly presents an overview on the use cases collected within the astronomical community by the IVOA Uptake committee.
- Section 3 describes how the Observation/Characterization data model interact and defines the core components of the Observation data model in UML. The various elements of the data model are summarized in

the table at figure 3.4.1

- Section 4 specifies the required data model fields as they are used in the TAP service: table names, column names, column datatype, utype from the Observation Core components data model, and required units.
- Section 5 provides information to register an Obs/TAP service into the Virtual Observatory Registries. More detailed description are available in the appendices
- Appendix A lists all the use cases as defined by the Take Up Committee
- Appendix C contains a full description of the Observation data model Core Components, that completes the table summary in data model section.
- Appendix D shows how to build up and fill in the TAP/SCHEMA tables necessary for the implementation of an OBS/TAP service.
- Appendix B shows examples of Obs/TAP queries.
- Section E describes two implementations realised with the Obs/Tap service
- Section F Updates of the document.

2 Use cases

We first focused on data discovery use-cases, aimed at finding observations in the VO domain, by broadcasting the same query to a bunch of data centers or to all VO subscribers.

Ultimately we need to provide data providers with a list of items and features that they could easily map to their database system, in order to answer to the kind of queries listed below.

The goal is to be simple enough to be implementable, and not to be exhaustive on all exotic data sets.

The main features of these use-cases are mainly:

- multi-wavelength search
- all kinds of data products(spectrum, cube, time series, etc.)
- various distribution formats data products(FITS, VOTable, compressed file archive,etc.)

Refined or advanced searches may include extra knowledge stemming from astronomical objects classification and would need to extract results from catalogs, possibly by using fine sub-queries.

The detailed list of use cases proposed for data discovery is in the appendix A.

3 Observation Core Components Data Model

This section highlights and describes the components of the observation data model necessary to support the well-identified set of use cases listed in Appendix A. That is, all parameters used to characterise the observations should be included in the model. In reality, it is worth mentioning here that this effort is the outcome of a trade-off between what Astronomers want and what Data Providers are ready to offer. The aim is to quickly buying-in data providers with a simple and good enough model to cover the majority of the Astronomers'needs.

The project of elaborating a general data model for the metadata necessary to describe any astronomical observation was launched at the first Data Model WG meeting held in Cambridge, UK at the IVOA meeting in May 2003. The Observation data model was sketched out relying on some key concepts : Dataset or Observation, Identification, Curation, Physical Characterisation and Provenance (either instrumental or software). A description of the early stages of this development can be found in [2], (Observation IVOA note). Some of these concepts have already been fleshed out in existing data models namely the Spectrum data model [3] for the general items and the Characterisation data model [4] for the description of the physical axes and properties of an observation.

3.1 Characterisation data model

The Characterisation data model organizes metadata as in a 3D matrix spanning independently the various physical axes of an observation (spatial, spectral, time, flux or whatever observable quantity), expressing 'Properties' such as coverage, resolution, sampling, accuracy of an observation along these axes and with up to four levels of description if necessary. This scheme allows to support selection of data sets for data discovery as well as data analysis.

3.2 Observation data model Core Components and attributes definition

Metadata of interest in the data discovery process are not only the one used to express queries and adjust selection criteria (type of product, positions, value, range, etc.) but also the returned parameters that allow to identify, in the query response the returned data sets and access to them.

From all the metadata covered and described in the full Observation data model, only a small part is needed to support data discovery in a regular and efficient manner. We then concentrate first on the definition of core

components of the Observation data model that will be used to support the use-cases described above. The Observation data model Core Components (ObsCore DM) is summarized in a UML class diagram below; this is a *logical* data model that described and binds together the main concepts related to an observation in astronomy. The implementation of such a model is described in Section 4.

3.2.1 UML description of the model

The Observation data model is organised according to some Object Oriented Programing principles in order to define unique and consistent concepts, as re-usable classes. UML helps to sketch out the class organisation as shown in Fig. 1. This class diagram covers all classes used in the context of the Observation Core components Model.

The Characterisation classes, describing how the data span along the main physical axes, are re-used here partly, with only the necessary attributes shown here. This is also the case for the DataID and Curation classes extracted from the Spectrum/SSA data model where only a subset of attributes are necessary for data discovery. We consider for now that we use Characterisation classes only down to the level of the Support Class (level 3).

For the sake of clarity, the SpatialAxis, SpectralAxis and TimeAxis classes on the diagram, are not expanded on the main class diagram.

Details for these axes are shown on Fig.2 for the Spatial axis, Fig.3 for the spectral axis.

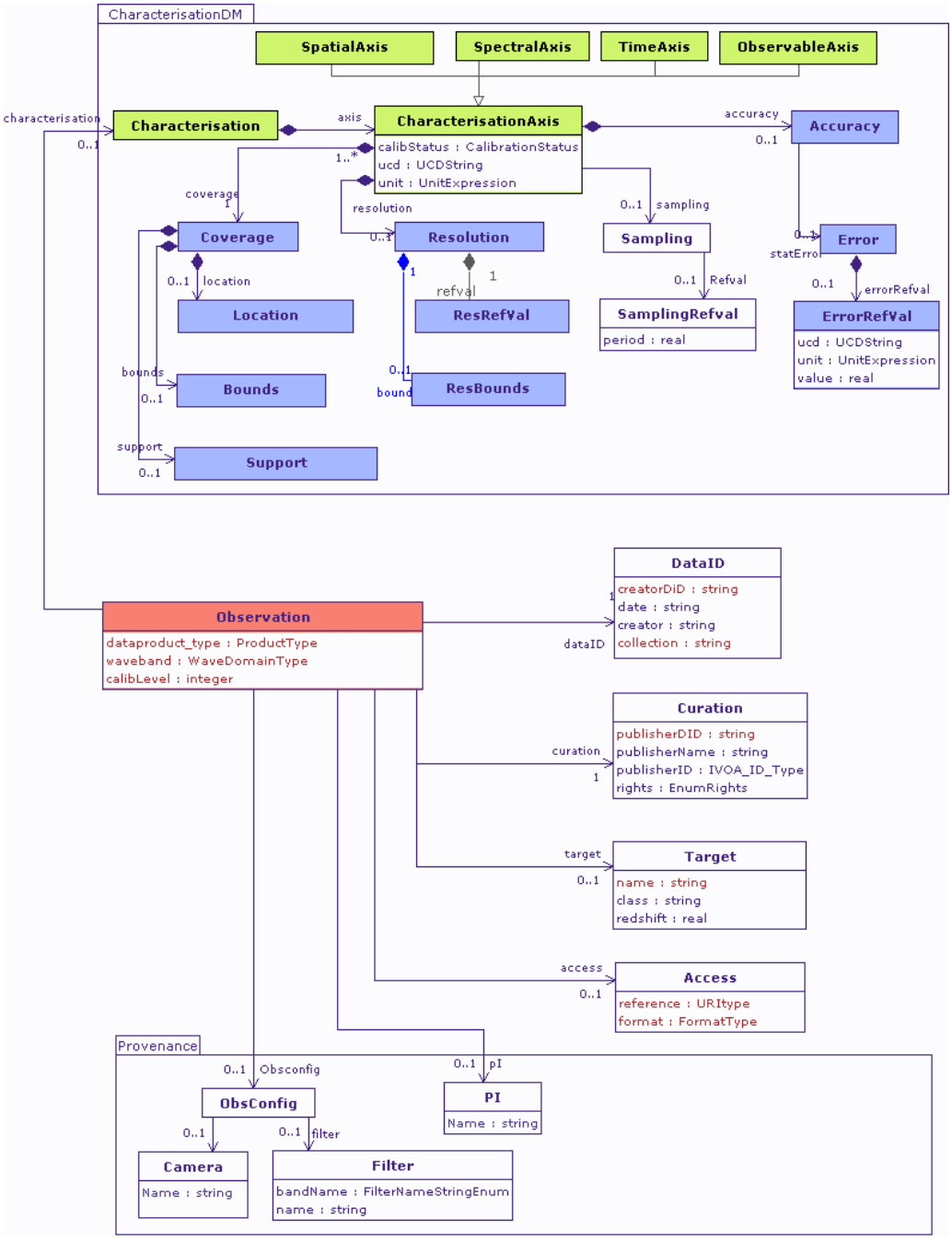

Figure 1: Here is a class diagram representing the classes used to organise observational metadata. Classes may be linked together via an association or aggregation link. The minimal set of necessary attributes for data discovery is shown in brown.

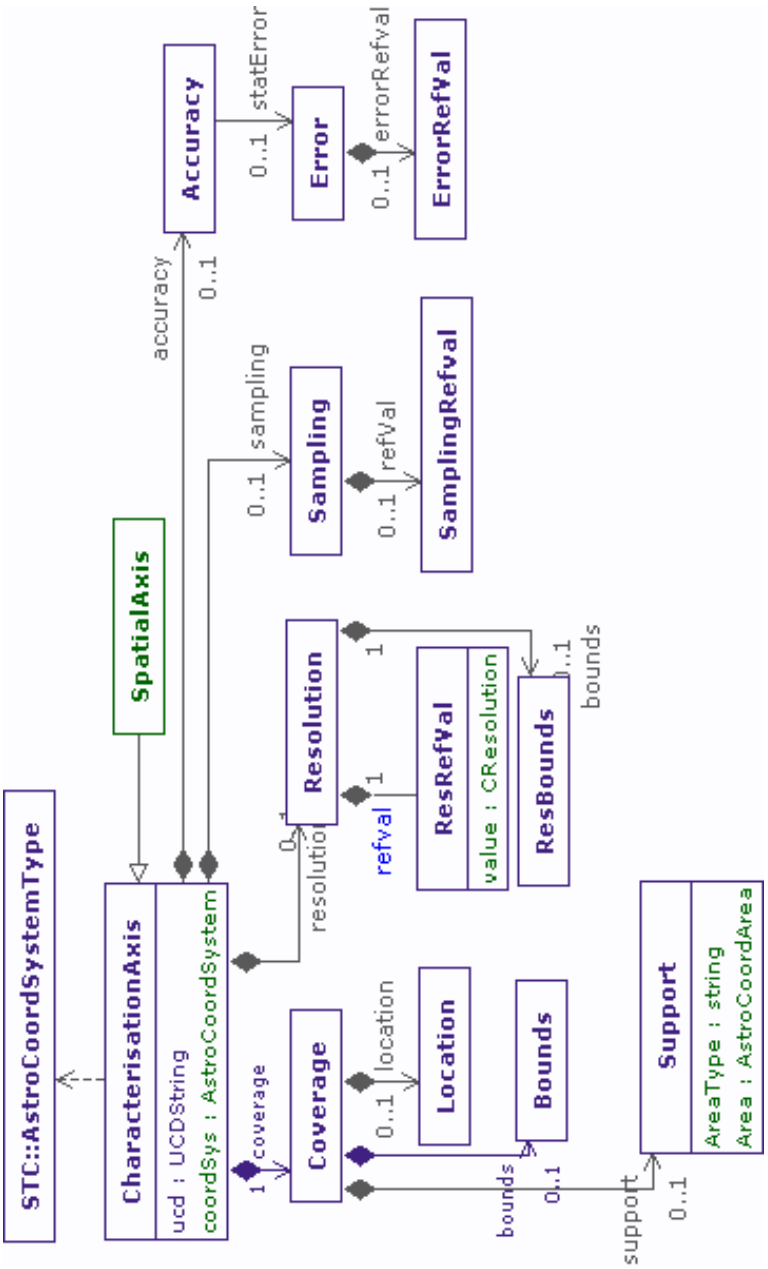

Figure 2: Details of the classes linked to the description of the Spatial axis for an Observation. All axes in this model inherits the main structure from the CharacterisationAxis class, but some peculiar attributes are necessary for Space coordinates.

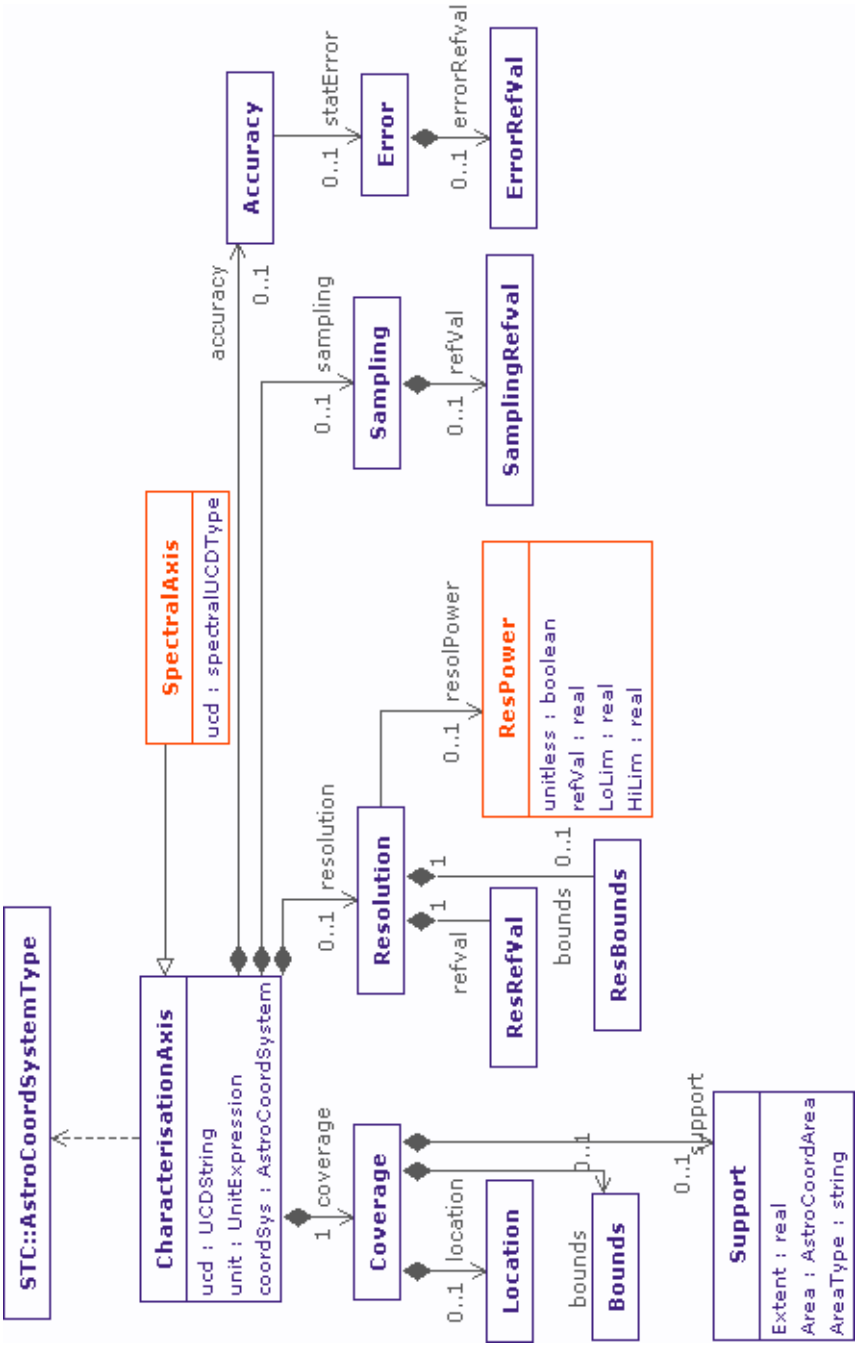

Figure 3: Spectral axis: details of the classes necessary to describe the spectral properties of an Observation. UCD and units are essential to disentangle various possible spectral quantities.

Details on the axes definition are available in the Chararacterisation data model standard document [4].

3.3 Data Model Summary

The table below contains a list of the fields defined in the Observation data model Core Components. The mandatory fields used in the an OBS/TAP service are listed in green ans labeled with **M**. They should be present in an Ivoa.ObsCore table for any service compliant to the Obs/Tap specification. See Section 4.

All other fields are optional and flagged with **O**; they can be used by data providers to support finer queries. These names are recommended only ; This gives the possibility to the data provider to define her/his own set of additional parameters, when not defined here. However, for the sake of homogeneity between services, using optional fields defined here is preferable.

The hypertext documentation of the model is (will be) available at `http://alinda.u-strasbg.fr/Model/ObsCore/index.html`.

The Appendix C contains an extensive description of the classes and attributes defined to capture all necessary metadata that complements the above list of items.

3.4 New concepts necessary for the Observation data model

Here we introduce new definitions in order to complete a description of a general observation, namely a data product type, for identifying the type of content, and a classification for various levels of calibration.

3.4.1 Type of Data Product

The model defines a *data product type* attribute for the Observation Class. It is the type of observation the user queries for or selects for retrieval. This is coded as a string that conveys a general idea of the content and organisation of a dataset. We consider a coarse classification of the types of dataset interesting for science usage, covering: image, spectrum, time series, visibility, event list, and cube. The Observation.DataProductType attribute takes its string value in the following set:

- **image**
- **spectrum**
- **cube**
- **timeseries**
- **visibility**
- **eventlist**

Observation Data model Summary. The core components are essentially the mandatory keys in the green cells of the table.

Local short name	Utype	Units	Type	Description	Status
OBSERVATION					
dataproduct_type	Observation.DataProductType	unitless	enum	Observation's type of product	M
calib_level	Observation.calibLevel	unitless	enum integer	Calibration level of the observation: in {0, 1, 2, 3}	M
target_name	Observation.Target.name	unitless	string	Object of interest	M
target_class	Observation.Target.Class	unitless	string	Class of the Target object as in SSA	O
DATAID					O
obs_id	Observation.DataID.collectionDID	unitless	string	internal ID given by the obs/tap service	M
obs_collection	Observation.DataID.Collection	unitless	string	Name of the data collection	M
creation_date	Observation.DataID.Date	unitless	date	Date when the data set was created	O
obs_creator_name	Observation.DataID.Creator	unitless	string	Name of the creator of the data	O
obs_creator_did	Observation.DataID.CreatorDID	unitless	string	Ivoa ID given by the creator	M
CURATION					
obs_publisher_did	Observation.Curation.PublisherDID	unitless	string	Data set ID given by the publisher.	M
publisher_id	Observation.Curation.PublisherID	unitless	string	ivoaID for the Publisher	O
bib_reference	Observation.Curation.Reference	unitless	string	Service bibliographic reference	O
data_rights	Observation.Curation.Rights	unitless	enum	Public/Reserved/Proprietary/	O
ACCESS					
access_url	Observation.Access.Reference	unitless	uri string	URL used to access dataset	M
access_format	Observation.Access.Format	unitless	string	format of the dataset: in {VOTable, FITS, FITS-EXT, Directory, etc..}	M
access_estsize	Observation.Access.Size	Kbytes	integer	Estimated Size of dataset: in Kilo Bytes	M
CHARACTERISATION					
nb_members	Characterisation.numseg	unitless	integer	Nb of obs. elements in a complex observation obtained by association, etc	O
space					
s_ra	Characterisation.SpatialAxis.Coverage.Location.coord.Position2D.Value2.C1	deg	double	Central Spatial Position in ICRS	M
s_dec	Characterisation.SpatialAxis.Coverage.Location.coord.Position2D.Value2.C2	deg	double		M
s_fov	Characterisation.SpatialAxis.Coverage.Bounds.Extent	deg	double	Estimated size of the covered region	M
s_region	Characterisation.SpatialAxis.Coverage.Support.Area		stc:AstroCoordArea	Region covered in STC or ADQL	M
s_resolution	Characterisation.SpatialAxis.Resolution.refVal.Cresolution	arcsec	float	Spatial resolution of data as FWHM	M

Local short name	Utype	Units	Type	Description	Status
s_ucd	Characterisation.SpatialAxis.ucd	unitless	ucd string	(pos or u,v data)	O
s_resolution_bound_min	Characterisation.Spatial.Resolution.bounds. Limits.Interval.LoLim	arcsec	double	Resolution min value on spectral axis (FHWM of PSF)	O
s_resolution_bound_max	Characterisation.Spatial.Resolution.bounds. Limits.Interval.LoLim	arcsec	double	Resolution max value on spectral axis	O
astrometric_cal_status	Characterisation.SpatialAxis.calibStatus	unitless	enum	NOT CALIBRATED, FINE, COARSE	O
astrom_precision_stat	Characterisation.SpatialAxis.Accuracy.StatError.refval.value	arcsec	double	Astrometric precision along the spatial axis	O
s_pixel_scale	Characterisation.SpatialAxis.Sampling.refVal.period	arcsec	double	Pixel spacing in spatial units	O
time					
t_min	Characterisation.TimeAxis.Coverage.Bounds.Limits.Interval.StartTime	day	double	Start time in MJD	M
t_max	Characterisation.TimeAxis.Coverage.Bounds.Limits.Interval.StopTime	day	double	Stop time in MJD	M
t_exptime	Characterisation.TimeAxis.Coverage.Support.Extent	s	float	Total exposure time	M
t_resolution	Characterisation.TimeAxis.Resolution.refVal	s	float	Temporal resolution FWHM	M
t_span	Characterisation.TimeAxis.Coverage.Bounds.Extent	day	float	Total observation elapsed time	O
t_cal_status	Characterisation.TimeAxis.calibStatus	unitless	enum	Type of coord calibration	O
t_staterr	Characterisation.TimeAxis.Accuracy.StatError.refval.value	s	double	Time coord statistical error	O
spectral					
em_min	Characterisation.SpectralAxis.Coverage.Bounds.limits.Interval.LoLim	m	double	start in spectral coordinates	M
em_max	Characterisation.SpectralAxis.Coverage.Bounds.limits.Interval.HiLim	m	double	stop in spectral coordinates	M
em_res_power	Characterisation.SpectralAxis.Resolution.ResolPower .refval	unitless	double	Value of the resolution power along the SpectralAxis.	M
em_resPower_min	Characterisation.Spectral.Resolution.ResolPower.LoLim	unitless	double	Resolution power min value on spectral axis	O
em_resPower_max	Characterisation.Spectral.Resolution.ResolPower.HiLim	unitless	double	Resolution power max value on spectral axis	O
em_resol	Characterisation.SpectralAxis.Resolution.refVal.value	m	double	Value of Resolution along the SpectralAxis	O
em_stat_err	Characterisation.SpectralAxis.Accuracy.StatError.refval.value	m	double	Spectral coord statistical error	O

Local short name	Utype	Units	Type	Description	Status
observable					O
o_ucd	Characterisation.ObservableAxis.ucd	unitless	string	Nature of the observable axis; necessary for polarisation data or any kind of flux. Values in { phot.flux, phot.flux.density, phot.count, phot.mag ,...}	M
o_units	Characterisation.ObservableAxis.units	unitless	enum	Units used for the observable values	O
o_cal_status	Characterisation.ObservableAxis.calibStatus	unitless	enum	Level of calibration for the observable coord	O
o_detection_limit	Characterisation.ObservableAxis.Resolution.refval	?	double	Minimal detectable value along the observable / senstitivity	O
o_stat_err	Characterisation.ObservableAxis.Accuracy.StatError.refval.value		double	Statistical error on the Observable axis	O
PROVENANCE					O
PI_name	Provenance.PI.name	unitless	string	Name of Principal Investigator	O
filter_band	Provenance.ObsConfig.Filter.bandName	unitless	string	For instance : U, B u, g i, k	O
filter_name	Provenance.ObsConfig.Filter.name	unitless	string	Filter name as stated into the archive: e.g FW66	O
camera_name	Provenance.ObsConfig.camera.name	unitless	string	Name of camera	O
optical_element_name	Provenance.ObsConfig.opticalElem.name	unitless	string	Name of optical element	O
telescope_name	Provenance.ObsConfig.telescope.name	unitless	string	Name of telescope	O
instrument	Provenance.ObsConfig.instrument.name	unitless	string	Name of the instrument used	O

3.4.2 Calibration level

The calibration level concept conveys to the users information on how much data reduction/processing has been applied to the data. It is up to the data provider to consider how to map his own internal classification to the suggested scale here:

- Level **0**: raw instrumental data, in proprietary or anyway internal provider format, that need specific tools to be handled.
- Level **1**: Instrumental data in a standard format(FITS, VOTable, SDFITS, ASDM, etc). Data sets with not all axes calibrated.
- Level **2**: Science ready data, with instrument signature removed, and calibration status defined on all physical axes.
- Level **3**: Enhanced data products like mosaics, improved co-added image cubes, resampled or drizzled images, etc., spectra with calibrated velocity axis at a particular line rest frequency.

Following examples can help to find the most appropriate value for the *calib_level* attribute.

3.4.2.1 Examples of data sets and their calibration level

Here are examples of various data sets, classified according to scheme defined above.

Editor MIR comment : please check . Should we add a link to some proper example files?

data product type	data collection name	Calibration Level	Comments
spectrum	XMM-Newton EPIC spectra	1	
Image	IRAS /NASA	2	science ready data
Image	IRIS/IRSA	3	recalibrated from infrared IRAS images with removal of the sensor memory effect.
Image	HDFS/ACS GOODS data	3	Image associations mosaicing/stacking
EventList	Rosat/HEASARC	1	
Visibility	Merlin	3	recombination wrt diff axes

Table 1: Exemples of data sets with calibration level

4 Implementation of ObsCore in a TAP Service

The ObsCore model will be implemented within Table Access Protocol (TAP) services such that all valid queries can be executed unchanged on any service that implements the model, as long as they do not make use of additional columns (see 4.20).

Here we specify an explicit mapping of the model to relational database tables; in the context of TAP this means we are specifying the logical tables as described in the `TAP_SCHEMA` (the TAP-required database schema where tables and columns included in the service are described). This does not necessarily imply that the underlying database will have the identical structure (what is exposed through TAP could be, for example, a database view of the underlying database tables), but in most cases the relationship between `TAP_SCHEMA` description and the underlying tables is straightforward.

schema_name	table_name	description
ivoa	ivoa.ObsCore	ObsCore 1.0

Table 2: TAP/SCHEMA.tables description of the ObsCore model

Tables 2 and 3 provides the primary information needed to describe the ObsCore model in terms of `TAP_SCHEMA` tables and columns. The "constraint" specified in Table 3 above is not part of the the `TAP_SCHEMA.columns` description, but is required by the ObsCore model and specified here to make this clear to implementors. Additional standard content for the individual columns is specified below. See Appendix B for a usable SQL script that inserts the standard `TAP_SCHEMA` content.

4.1 Data Product Type

- `column_name`: `dataproduct_type`
- datatype: adql:VARCHAR
- size: implementor decides
- units: NULL
- utype: Observation.DataProductType
- UCD: meta.id;class
- principal: 1
- indexed: implementor decides
- std: 1

The `dataproduct_type` column contains a simple string value describing the primary nature of the data product. Allowed values are: image, spectrum, timeseries, visibility, eventlist, cube. Values are in lower case. Subtypes, separated by one or more dots from the base categorisation, may be specified in provider-specific columns (see 4.20).

Values in the `dataproduct_type` column must not be NULL.

TODO: reconsider extending to two-level product type when the data model in previous section is completed.

table_name	column_name	datatype	units	constraint
ivoa.ObsCore	dataproduct_type	adql:VARCHAR		not null
ivoa.ObsCore	calib_level	adql:INTEGER		not null
ivoa.ObsCore	obs_collection	adql:VARCHAR		not null
ivoa.ObsCore	obs_id	adql:VARCHAR		not null
ivoa.ObsCore	obs_publisher_did	adql:CLOB		not null
ivoa.ObsCore	access_url	adql:CLOB		
ivoa.ObsCore	access_format	adql:VARCHAR		
ivoa.ObsCore	access_estsize	adql:INTEGER	KB	
ivoa.ObsCore	target_name	adql:VARCHAR		
ivoa.ObsCore	s_ra	adql:DOUBLE	deg	
ivoa.ObsCore	s_dec	adql:DOUBLE	deg	
ivoa.ObsCore	s_fov	adql:DOUBLE	deg	
ivoa.ObsCore	s_region	adql:REGION	deg	
ivoa.ObsCore	s_resolution	adql:DOUBLE	arcsec	
ivoa.ObsCore	t_min	adql:DOUBLE	d	
ivoa.ObsCore	t_max	adql:DOUBLE	d	
ivoa.ObsCore	t_exptime	adql:DOUBLE	s	
ivoa.ObsCore	t_resolution	adql:DOUBLE	s	
ivoa.ObsCore	em_min	adql:DOUBLE	m	
ivoa.ObsCore	em_max	adql:DOUBLE	m	
ivoa.ObsCore	em_res_power	adql:DOUBLE		
ivoa.ObsCore	o_fluxucd	adql:VARCHAR		

Table 3: TAP/SCHEMA.columns description of the ivoa.ObsCore table

4.2 Calibration Level

- `column_name`: `calib_level`
- datatype: adql:INTEGER
- size: NULL
- units: NULL
- utype: Observation.calibLevel
- UCD: meta.id;class
- principal: 1
- indexed: implementor decides
- std: 1

The `calib_level` column tells the user the amount of calibration processing that has been applied to create the data product. Allowed values are: 0 (instrumental/raw data in a non-standard format), 1 (raw data in a standard format), 2 (calibrated data in standard format, instrument signature removed), and 3 (more highly processed data product). Data providers decide which value best describes their data products.

Values in the `calib_level` column must not be NULL. TODO Examples of calib level for well known data set examples:

4.3 Collection Name

- `column_name`: `obs_collection`
- datatype: adql:VARCHAR
- size: implementor decides
- units: NULL
- utype: Observation.DataID.collection
- UCD: meta.id
- principal: 1
- indexed: implementor decides
- std: 1

The `obs_collection` column identifies the data collection to which the data product belongs. A data collection can be any collection of datasets which are alike in some fashion. Typical data collections might be all the data from a particular telescope, instrument, or survey. The value is either the registered shortname for the data collection, the full registered IVOA identifier for the collection, or a data provider defined shortname for the collection. Examples: HST/WFPC2, VLT/FORS2, CHANDRA/ACIS-S, etc.

Values in the `obs_collection` column must not be NULL.

4.4 Observation Identifier

- `column_name`: `obs_id`
- datatype: adql:VARCHAR
- size: implementor decides
- units: NULL
- utype: Observation.DataID.CreatorDID
- UCD: meta.id
- principal: 1
- indexed: implementor decides

- std: 1

The `obs_id` column contains a collection-specific identifier for an observation. In the case where multiple data products are available for an observation (e.g. with different calibration levels), the `obs_id` value will be the same for each product of the observation.

Values in the `obs_id` column must not be NULL.

4.5 Publisher Dataset Identifier

- `column_name`: `obs_publisher_did`
- datatype: adql:CLOB
- size: implementor decides
- units: NULL
- utype: Observation.Curation.PublisherDID
- UCD: meta.ref.url;meta.curation
- principal: 1
- indexed: implementor decides
- std: 1

The `obs_publisher_did` column contains the IVOA dataset identifier [7] for the published data product. This value is globally unique since different publishers (data centres) that provide access to a data product will each assign their own value.

We specify the datatype as CLOB in the TAP service so that users will know they can only use the `obs_publisher_did` column in the select clause of a query. That is, users cannot specify this column as part of a condition in the WHERE clause and implementors are free to generate the URL during output.

Values in the `obs_publisher_did` column must not be NULL.

4.6 Access URL

- `column_name`: `access_url`
- datatype: adql:CLOB
- size: NULL
- units: NULL
- utype: Observation.Access.Reference
- UCD: meta.ref.url
- principal: 1
- indexed: 0
- std: 1

The `access_url` column contains a URL that can be used to download the data product (files).

We specify the datatype as CLOB in the TAP service so that users will know they can only use the `access_url` column in the SELECT clause of a query. That is, users cannot specify this column as part of a condition in the WHERE clause and implementors are free to generate the URL during output.

4.7 Access Format

- `column_name`: `access_format`

- datatype: adql:VARCHAR
- size: NULL
- units: NULL
- utype: Observation.Access.Format
- UCD: meta.id;class
- principal: 1
- indexed: 0
- std: 1

The `access_format` column contains a string indicating the format of downloaded the data product (files). The values are made up of dot-deparated words that describe (in increasing detail) the structure of the data file(s). Allowed values include any legal mimetype [REF] that describes the format or one of the following short forms: fits, fits.image, fits.image.mef, fits.table, fits.cube, fits.cube.mef, directory, VOTable, csv, tsv, pdf. TODO: Doug and Arnold to provide list by 2010-06-15.

4.8 Estimated Download Size

- `column_name`: `access_estsize`
- datatype: adql:BIGINT
- size: NULL
- units: KB
- utype: Observation.Access.Size
- UCD: size??
- principal: 1
- indexed: 0
- std: 1

The `access_estsize` column contains an approximate size (in KB) of the file(s) available via the `access_url`.

4.9 Target Name

- `column_name`: `target_name`
- datatype: adql:VARCHAR
- size: implementor decides
- units: NULL
- utype: Observation.Target.Name
- UCD: meta.id;src
- principal: 1
- indexed: implementor decides
- std: 1

The `target` column contains the name of the target of the observation. This is typically the proper name of an astronomical object, but could be the name of a survey field.

4.10 Central Coordinates

The coordinate system in which coordinates are expressed is by default ICRS.

- `column_name`: `s_ra`
- datatype: adql:DOUBLE
- size: NULL
- units: deg
- utype: Characterisation.SpatialAxis.Coverage.Location.coord.Position2D.Value2.C1
- UCD: pos.eq.ra
- principal: 1
- indexed: implementor decides
- std: 1

The `s_ra` column stores the ICRS Right Ascension of the centre of the observation.

- `column_name`: `s_dec`
- datatype: adql:DOUBLE
- size: NULL
- units: deg
- utype: Characterisation.SpatialAxis.Coverage.Location.coord.Position2D.Value2.C2
- UCD: pos.eq.dec
- principal: 1
- indexed: implementor decides
- std: 1

The `s_dec` column stores the ICRS Declination of the centre of the observation.

4.11 Spatial Extent

- `column_name`: `s_fov`
- datatype: adql:DOUBLE
- size: NULL
- units: deg
- utype: Characterisation.SpatialAxis.Coverage.Bounds.Extent
- UCD: phys.angSize;instr.fov
- principal: 1
- indexed: implementor decides
- std: 1

The `s_fov` column contains the approximate size of the region covered by the data product. For a circular region, this is the diameter (not the radius). It is an estimate of the Aperture angular size , as used in the Spectrum DM.

4.12 Spatial Coverage

- `column_name`: `s_region`
- datatype: adql:REGION
- size: NULL
- units: deg
- utype: Characterisation.SpatialAxis.Coverage.Support.Area
- UCD: ?
- principal: 1
- indexed: implementor decides

- std: 1

We specify the datatype as the logical type REGION so that users can specify spatial queries using a single column and in a limited number of ways. If included in the select list of the query, the output is always an STC-S string as described in [1][section 6]. In the WHERE clause, the `s_region` column can be used with the ADQL geometry functions (INTERSECTS, CONTAINS) to specify conditions; the service will generally have to translate these into native SQL that enforces the same conditions or a suitable approximation. Implementors may approximate the spatial query conditions by translating the INTERSECTS and CONTAINS function calls in the query.

In addition, ADQL specifies several functions which may take the `s_region` column as an argument: AREA, CENTROID, and COORDSYS. The AREA function returns the area (in sq. deg.) of the observed the region. In cases where the `s_region` itself is an approximation (a bounding box, for example), this function should still return the actual value. This may be implemented by computing and storing the area in a separate column and converting the AREA(`s_region`) function call into a column reference in the query. The CENTROID function returns an ADQL POINT value; if used in the select list the output is always an STC-S string as described in [1][section 6]. The coordinates must be the same as those found in the `s_ra` and `s_dec` columns, which are provided for convenience. The COORDSYS function returns the coordinate system used for the `s_region`; in the ObsCore model implementation here this is restricted to ICRS, so this can be implemented by converting the COORDSYS(`s_region`) function call to a constant in the query.

4.13 Spatial Resolution

- `column_name`: `s_resolution`
- datatype: adql:DOUBLE
- size: NULL
- units: arcsec
- utype: Characterisation.SpatialAxis.Resolution.refVal.CResolution
- UCD: pos.angResolution
- principal: 1
- indexed: implementor decides
- std: 1

TODO: define here...

4.14 Time Bounds

- `column_name`: `t_min`
- datatype: adql:DOUBLE
- size: NULL
- units: d
- utype: Characterisation.TimeAxis.Coverage.Bounds.limits.Interval.StartTime
- UCD: time.start;obs.exposure
- principal: 1
- indexed: implementor decides
- std: 1

The `t_min` column contains the start time of the observation in Modified Julian Day(s).

- `column_name`: `t_max`
- datatype: adql:DOUBLE
- size: NULL
- units: d
- utype: Characterisation.TimeAxis.Coverage.Bounds.limits.Interval.StopTime
- UCD: time.stop;obs.exposure
- principal: 1
- indexed: implementor decides
- std: 1

The `t_max` column contains the stop time of the observation in Modified Julian Day(s).

4.15 Exposure Time

- `column_name`: `t_exptime`
- datatype: adql:DOUBLE
- size: NULL
- units: sec
- utype: Characterisation.TimeAxis.Coverage.Support.Extent
- UCD: time.duration;obs.exposure
- principal: 1
- indexed: implementor decides
- std: 1

The `t_exptime` column contains the exposure time. For simple exposures, this is just `t_max - t_min` expressed in seconds. For data where the detector is not active at all times (e.g. products made by combining exposures taken at different times), the `t_exptime` will be smaller than `t_max - t_min`. For data where the `t_exptime` is not constant over the entire data product, the median exposure time per pixel is a good way to characterise the typical value. In all cases, `t_exptime` is generally used as an indicator or relative sensitivity (depth) within a single collection (e.g. `obs_collection`); providers should supply a suitable relative value when it is not feasible to define or compute the true exposure time.

4.16 Time Resolution

- `column_name`: `t_resolution`
- datatype: adql:DOUBLE
- size: NULL
- units: sec
- utype: Characterisation.TimeAxis.Resolution.refVal
- UCD: time.resolution
- principal: 1
- indexed: implementor decides
- std: 1

The minimal interpretable interval between two points along the time axis. Can be an averaged or typical value.

4.17 Spectral Bounds

As mentioned in the data model appendix, C.5.1.3, at least 3 physical quantities can be mapped on the spectral axis: energy, wavelength and frequency. In this simple model we restrict the spectral bounds to wavelength expressed in meter units. Conversion to other quantities could be performed either at the client site , for an application encapsulating queries, or at the server side , for a data provider to expose its data from other regimes to OBS/TAP queries.

- `column_name`: `em_min`
- datatype: adql:DOUBLE
- size: NULL
- units: m
- utype: Characterisation.SpectralAxis.Coverage.Bounds.limits.Interval.LoLim
- UCD: em.wl;stat.min
- principal: 1
- indexed: implementor decides
- std: 1

The `em_min` column contains the minimum energy observed, expressed as (Barycentric? vacuum?) wavelength.

- `column_name`: `em_max`
- datatype: adql:DOUBLE
- size: NULL
- units: m
- utype: Characterisation.SpectralAxis.Coverage.Bounds.limits.Interval.HiLim
- UCD: em.wl;stat.max
- principal: 1
- indexed: implementor decides
- std: 1

The `em_max` column contains the maximum energy observed, expressed as (Barycentric? vacuum?) wavelength.

4.18 Spectral Resolution (Resolving Power)

- `column_name`: `em_res_power`
- datatype: adql:DOUBLE
- size: NULL
- units: NULL
- utype: Characterisation.SpectralAxis.Resolution.ResPower.refVal
- UCD: spec.resolution
- principal: 1
- indexed: implementor decides
- std: 1

The `em_res_power` column contains the typical or characteristic resolving power of the energy axis. The value is dimensionless (e.g. delta lambda /lambda).

4.19 Observable Axis Description

- `column_name`: `o_fluxucd`
- datatype: adql:VARCHAR
- size: implementor decides
- units: NULL
- utype: Characterisation.ObservableAxis.ucd
- UCD: meta.code
- principal: 1
- indexed: implementor decides
- std: 1

The `o_fluxucd` column contains a UCD [9] describing the nature of the observable within the data product. The observable is the measured quantity, for example photon counts or photon density stored in the pixel value within an image.

TODO: Doug and Arnold to provide a list of observables, Francois and Mireille to map this to a list of typical UCDs by 2010-06-15.

4.20 Additional Columns

Service providers may include additional columns in the ivoa.ObsCore table to expose additional metadata. These columns must be described in the `TAP_SCHEMA.columns` table and in the output from the VOSI-tables resource. Users may access these columns by examining the column metadata for individual services and then using them explicitly in queries or by selecting all columns in the query (e.g. "select * from ivoa.ObsCore ..." in an ADQL query). In order to provide homogeneity in the keywords used as optional fields, we recommend to use the items defined in the data model and flagged as optional.

5 Registering a TAP Service with the ObsCore Model

TAP services that implement the ObsCore model should be registered to indicate this fact so that users can easily find all services that accept ObsCore queries. Initially, this can be done by using the keyword "ObsCore" to describe the service. Fine-grained registries may include the complete VODataService `tableset` description, but this is not available through all searchable registries. The TAP extension schema [to come] allows service providers to specify which IVOA data models are used within a TAP service. Services that implement the ObsCore model should use the registry extension schema mechanism to publicise their support with:

```
standardID="ivo://ivoa.net/std/ObsCore/v1.0"
```

placed (somewhere) in the TAP Registry extension schema [REF], where `ivo://ivoa.net/std/ObsCore/v1.0` is the IVOA identifier of the StandardRegExt [REF] defined in this document.

References

[1] Dowler P. et al. Data model for astronomical dataset characterisation. *http://www.ivoa.net/Documents/latest/IDs.html*, 2007.

[2] J. McDowell et al. Observation data model for astronomical dataset. *http://www.ivoa.net/Documents/latest/DMObs.html*, 2005.

[3] Jonathan McDowell et al. Ivoa spectral data model. *http://www.ivoa.net/Documents/latest/SpectrumDM.html*, 2007.

[4] Louys M. et al. Data model for astronomical dataset characterisation. *http://www.ivoa.net/Documents/latest/CharacterisationDM.html*, 2007.

[5] Michel L. et al. Xmm newton catalog service xcatdb, 2008.

[6] Michel L. et al. Database construction for heterogeneous data sets with saada, 2009.

[7] Plante R. et al. Ivoa identifiers. *http://www.ivoa.net/Documents/latest/IDs.html*, 2007.

[8] Plante R. et al. Vodataservice : a voresource schema extension for describing collections and services. *http://www.ivoa.net/Documents/latest/VODataService/*, 2010.

[9] Preite Martinez A. et al. The ucd1+ controlled vocabulary. *http://www.ivoa.net/Documents/latest/UCDlist.html*, 2007.

[10] DAL WG. Simple spectral access protocol. *http://www.ivoa.net/Documents/latest/SSA*, 2007.

A Use Cases in detail

The ability to discover data of a certain kind (*images, spectra, cubes, etc.*) according to some scientific criteria (*e.g., a given sky position, spectral coverage including spectral line X, spatial resolution better than Y, resolving power greater than Z*) is central to archival Astronomy. A special Take Up Committee of the IVOA was formed in 2009 to stimulate the IVOA work in the area of a catalogue-based data access layer that allows astronomers to easily query and access scientific data. Such committee came up with a list of data discovery use cases expressed as a set of constraints on selected scientific parameters to be met by the datasets of interest. The full list of use cases is herein included.

A.1 Use Cases 1: Discover imaging data of interest

Use case 1.1: Show me a list of all data that satisfies

i DataType=any
ii Energy includes 5 keV
iii RA includes 16.00
iv DEC includes +10
v Exposure time > 10 ks

Use case 1.2: Let me input a list of RA and DEC coordinates and show me spatially coincident data that satisfies

i Imaging or spectroscopy data
ii Includes one or more of the RA,DEC values in the list (LIST=SERVICE REQ)
iii Includes both a wavelength in the range 5000-900 angstroms AND an X-ray image (AND=SERVREQ)

Use case 1.3: Show me a list of all data that satisfies

i DataType=Image
ii Spatial resolution better than 0.3 arcseconds
iii Filter = J or H or K
iv RA between 16 hours and 17 hours
v DEC between 10 degrees and 11 degrees

Use case 1.4: Show me a list of all data that satisfies

i DataType=Image
ii Wavelength=V or I or Z
iii Spatial Resolution < 0.7 arcseconds FWHM
iv Exposure Time > 1000 seconds
v Data Quality: Fully Calibrated

Use case 1.5: Show me all data that satisfies

i DataType=IFU
ii DataQuality: Fully Calibrated
iii ObjectClass=quasar (SERVIC REQ + NEEDS ANOTHER SERVICE (CATALOGUE)
iv Redshift > 3.
v Radioflux > 1 mJy

Use case 1.6: For an input list of RA,DEC, Mean Julian Date, show me all data that satisfies (LIST=SERVREQ)

i DataType=imaging
ii RA,DEC include the value and the list and Observation date is within 1 day of the MJD value

A.2 Use Cases 2: Discover spectral data of interest

Use case 2.1: Show me a list of all data that satisfies

i DataType=Spectrum
ii Energy spans 1 to 5 keV
iii Total counts in spectrum > 100
iv Exposure time > 10000 seconds
v Data Quality: Fully Calibrated

Use case 2.2: Show me a list of all data that satisfies

i DataType=Spectrum
ii Wavelength includes 6500 angstroms
iii Spectral Resolution better than 15 angstroms
iv Spatial Resolution better than 2 arcseconds FWHM
v Exposure Time > 3600 seconds
vi Data Quality = Any

Use case 2.3: Show me a list of all data that satisfies

i Emission line width $H_\alpha > 2000$ km/s FWHM (SERVICEREQ+NEEDS OTHER SERVICE)
ii $H_\alpha / H_\beta > 3.5$

A.3 Use Cases 3: Discover data cubes of interest

Use case 3.1: Show me a list of data

i DataType=cube (IFU spectroscopy?)
ii RA,DEC includes value RA1,DEC1
iii Field size > 100 square arcseconds
iv DataSensitivity = deep
v Spectral resolution better than 5 angstroms FWHM

Use case 3.2: Show me a list of all data that satisfies

i DataType=Cube with 3 dimensions
ii Axes includes Velocity
iii Axes includes RA
iv Axes includes DEC
v Velocity Resolution better than 50 km/s
vi RA includes 16.000
vii Dec includes +41.000

Use case 3.3: Show me a list of all data that satisfies

i DataType=cube

ii RA includes 16.00
iii Dec includes +41.00
iv Wavelength includes 6500 angstroms
v Wavelength includes 4000 angstroms
vi Spectral resolution better than 5 angstroms
vii Exposure time more than 3600 seconds
viii Data Quality: Fully Calibrated

Use case 3.4:Show me a list of all data that satisfies

i DataType=Cube with 3 dimensions
ii Axes includes FREQ
iii Axes includes RA
iv Axes includes DEC
v Velocity Resolution better than 1 km/s
vi RA includes 83.835000
vii Dec includes -5.014722
viii Rest Frequency = 345.795990 GHz
ix VLSRK in the range [6.0, 10.0]

Use case 3.5:Show me a list of all data that satisfies

i DataType=Cube with 3 dimensions
ii Axes includes FREQ
iii Axes includes RA with $>$ 100 pixels
iv Axes includes DEC with $>$ 100 pixels
v Frequency extent $>$ 500 MHz
vi Rest Frequency = 345.795990 GHz appears in band
vii The redshift is not specified, but should default to z_{source} for the target.

Use case 3.6:Show me a list of all data that satisfies

i DataType=Cube with 3 dimensions
ii Axes includes FREQ
iii Axes includes RA
iv Axes includes DEC
v Frequency resolution $<$ 10 MHz
vi Rest Frequency = 337.2966 GHz appears in band
vii Any observation that could have detected a line at this rest frequency from any target, using the nominal redshift for the target.

Use case 3.7:Show me a list of all data that satisfies

i DataType=Cube with 3 dimensions
ii Axes includes FREQ
iii Axes includes RA
iv Axes includes DEC
v Frequency resolution $<$ 10 MHz
vi Rest Frequency in (213.36053, 256.0278, 298.6908925, 341.350826, 384.0066819, 426.6579505, 469.3041221, 511.944687, 554.5791355) GHz appears in band
vii Any observation that could have detected HCS+ (list of transition rest frequencies given above) from any target, using the nominal redshift for the target.

Use case 3.8:Show me a list of all data that satisfies

i DataType=Cube with 4 dimensions
ii Axes includes FREQ
iii Axes includes RA with > 100 pixels
iv Axes includes DEC with > 100 pixels
v Axes includes STOKES
vi Frequency resolution < 1 MHz
vii Rest Frequency $= 345.795990$ GHz appears in band

Use case 3.9: Looking for moving targets

i Show me the names of all the objects that have moving coordinates (i.e. no RA,Dec position).

A.4 Use Cases 4: Discover time series of interest

Use case 4.1: Show me a list of all data that satisfies

i DataType=TimeSeries
ii RA includes 16.00 hours
iii DEC includes +41.00
iv Time resolution better than 1 minute
v Time interval (start of series to end of series) > 1 week
vi Observation data before June 10, 2008
vii Observation data after June 10, 2007

A.5 Use Cases 5: Discover general data of interest

Use case 5.1: Show me a list of all data that satisfies

i Optical imaging
ii In the M81 group
iii With area greater than 0.5 degress square
iv With sensitivity $> 10 * \sigma$ for point source m=25
v I also want X-ray data with cutouts 5 arcmin on a side of all the detected galaxies
vi I also want Radio data cutouts 5 arcmin on a side around detected galaxies

Use case 5.2: Show me a list of all data that satisfies

i DataType=Imaging or Spectroscopy
ii RA includes 16.00 hours
iii DEC includes +41.00 degress
iv SDSS images and spectra AND CFHTLS images and spectra

Use case 5.3: In Virgo cluster show me imaging and X-ray data for all galaxies that are cluster members and have $B < 21$

A.6 Use Cases 6: Others

6.1 Given COSMOS (or other survey) X-Ray source catalogue give me all the sources with $photoZ > X$, and spiral galaxy counterpart and produce radio - to -X-ray SEDs

6.1 Comment Requires source/object catalogues to drive data query (for SED info which may be catalogue or data)

6.2 Given a list of Abell clusters, give me all their Chandra images with $texp > X$, after I select regions occupied by the diffuse emission, give me all the Chandra point sources in these regions, and find their redshift (I want to find background quasars because I am interested in lensing and I have no idea where to go to find z). For the quasars, give me high res (< 0.5") optical and radio images, and build SEDs

6.2 Comment Requires source/object catalogues and interactive image interactions (applications/interfaces), further query, and more catalogues to drive data query.

6.3 Find me all the variable Chandra sources with optical counterpart and redshift. If redshift is not available, give me an SED to compare with source templates (I also would like to run a tool or obtain a library of such templates from a theory database, which I expect the VO to provide). My aim is to separate stars from variable quasars.

6.3 Comment Pretty complicated, including templates and theory as well as catalogues.

B Some Use cases translated as Obs/TAP ADQL queries

Here we consider a very general use case from the list (1.1) :
Show me a list of all data that satisfies:

1.1.1 Datatype=any
1.1.2 contains RA=16.0 and DEC=40.0

These data would be searched on all VO services by sending the following query:

```
SELECT * FROM ivoa.Obscore WHERE
CONTAINS(POINT(16.0,40.0),s_region)
```

More constraints can be added in the following use-case (1.3):
Show me a list of all data that satisfies

1.3.1 DataType=Image
1.3.2 Spatial resolution better than 0.3 arcseconds
1.3.3 Filter = J or H or K
1.3.4 RA between 16 hours and 17 hours
1.3.5 DEC between 10 degrees and 11 degrees

Such a query needs to compute RA in degrees, extract information from Filter and adjust spectral intervals for search.

```
SELECT * FROM ivoa.Obscore_ext WHERE
dataproduct_type=‘‘Image.2D’’
AND  s_resolution < 0.3
AND s_ra > 240 AND s_ra < 255 AND
s_dec > 10 AND s_dec < 11
AND
(em_min > 2.1 AND em_max < 2.4) OR
(em_min >= 1.6 AND em_max <= 1.8) OR
(em_min >= 1.2 AND em_max <= 1.4)
```

C Data Model detailed description

Here is a full description of the concepts used in the model, the classes and attributes used to support them. We also mention the corresponding "short name" that would be used in the OBS/TAP implementation of this model and is part of Table 3.4.1. The prefix for all utypes in the Observation Core Components DM is ***'obs:'*** but has been eluded here for a simpler reading.

C.1 Observation

This class is a place holder that gathers all metadata relative to an observed and distributed dataset. It points to existing classes of Spectrum DM and types from VODataservice [8].

C.1.1 Type of Observation

`dataproduct_type`

The model defines a *data product type* attribute for the Observation Class. It is the type of observation the user queries for or selects for retrieval. This is coded as a string that conveys a general idea of the content and organisation of a dataset. Possible values for this string are described in section 3.4. The short name for this attribute is `dataproduct_type` in the implemented Ivoa.ObsCore table, as shown in Table 3.

C.1.2 Calibration level

`calib_level`

It is a convention we suggest to use to classify the different possible calibration status of an observed dataset. These 4 categories allow to distinguish 4 levels of calibration and would be sufficient for 80 % of the data collections. This will be up to the data provider to consider how to map his own internal classification to the suggested scale here.

Following examples can help to find the most appropriate value for the *calibLevel* attribute.

- Level 0:
  raw instrumental data, possibly in proprietary internal provider format, that need specific tools to be handled.
- Level 1:
  Instrumental data in a standard format(FITS, VOTable, SDFITS, ASDM, etc. The data may or may not be calibrated. Standards tools can handle it.
- Level 2:
  Science ready data , with instrument signature removed, and calibration status defined on all physical axes.
- Level 3:
  Enhanced data products like mosaics, improved co-added image cubes, resampled or drizzled images, etc. spectra with calibrated velocity axis at a particular line rest frequency. In such case, the improved calibration procedure is described by the data provider in some way, progenitors of such a data product can be identified into the reduction pipeline.

This classification is simple enough to cover all regimes. Data providers will adjust the mapping of their various internal levels of calibration to this general frame, with the knowledge of the PIs for each project.

C.1.3 Target

This is the astronomical object of interest for which the observation was performed. Only the name is used in the Core Components model but the Target object is fully designed in the Spectrum data model. Serendipitous archives or surveys may not contain this information for all observations, so this can be 'NULL' if necessary or have a value like 'DARK' for instance to specify a dark field and not a pointed observation.

C.1.3.1 Target Name `target_name`

The first attribute is the name of the target and is flagged as mandatory.

C.1.3.2 Class of the Target source/object `target_class`

Indicates the type of object that was pointed for this observation. It is a string with possible values defined in a special vocabulary set to be defined : list of object classes (or types) used by the SIMBAD database, f.i., or defined in another IVOA vocabulary.

A simple set can be defined for this data model version : STAR, GALAXY, QUASAR, RADIO SOURCE, LINER, GLOB.CLUSTER, DARK, OTHER
TO BE completed

C.2 Observation regime

`em_domain`

The spectral regime to which an observation belongs is modeled as *Observation.waveband* with short name : `em_domain` . This re-uses the waveband definition from the VODataService IVOA standard and the same set of enumerated strings: RADIO, MILLIMETER, INFRARED, OPTICAL, UV, EUV, X-RAY, GAMMA-RAY available at [8].

C.3 Identification/DataID Class

After acquisition and reduction an observation is uniquely identified by its creator and gets a creator dataset identifier. This information is defined in the Spectrum data model in the DataID class. We re-use this class and the 'Observation.DataID.CreatorDID' Utype string in order to distinguish two dataset curated by two different services (archives) but originating from the same creator. When broadcasting a query to multiple servers, the response may contain multiple copies of the same dataset, with a unique `obs_creator_did` but possibly different `obs_publisher_did` (given by the data provider). Therefore a unique identifier is needed here. In the ObsCore model, the short name associated to this ID should be `obs_id` (to be checked).

The second identifier used in this model is the one given by the data provider, and defined in the Curation Class.

C.3.1 Creator name

`creator_name`
This parameter gives the name of the institution which created the dataset.

C.3.2 Creation date

`creation_date`
The creation date parameter gives the date when the dataset has been created.

C.4 Curation metadata

The Curation Class inherits from the Spectrum data model and VOResource concepts too. The various attributes for ObsCore are:

C.4.0.1 Publisher Dataset ID `publisher_did`
This is the identifier the publisher provides for this observation . It may differ from the original id given by the creator of the data set. (new reduction , new version , etc..). The corresponding Utype mapped from Spectrum DM is Observation.Curation.PublisherDID and relates to the same definition as in the Resource Metadata standard.

C.4.0.2 Publisher Name `publisher_name`
The name of the service providing data : archive, data center, etc.. Not fully standardised. The corresponding Utype is Observation.Curation.Publisher.

C.4.0.3 Publisher identifier `publisher_id`
The IVOA ID for the data provider.

C.4.0.4 Bibliographic reference `bib_reference`
Url or Bibcode for documentation. This is a forward link to publications which reference the data set.This is an exact re-use of the SSA definition. See [10] Curation Metadata 4.2.5.6

C.4.0.5 Data Rights `data_rights`
This parameter allows to mention protected, public and not yet released data. The date of the future release can be added for data set in the PI proprietary period. Possible values are : PUBLIC, PROTECTED, PROPRIETARY (check see VODataService? or VOresource)

C.5 Number of observed segments(or components)

`num_seg` Although we mostly deal with single observations, some data products are the result of combinations of multiple original observations. This is the case for the SED of a source, for drizzled, co-added, mosaic images for instance. This will also be the case for IFU data cubes obtained by summing dozens of short poses, etc.

This field provides the number of original progenitors used to build up the observation of interest. It is stored as an integer value. This information is necessary for a full

description of the characterisation of complex data products, but also from a provenance point of view.This clearly belongs to the features of an Observation but can be re-used within the Characterisation class of a global complex result of an observation combination process. How this field is shared between the two classes will be defined in a new revision (v 2.0) of the Characterisation data model. The corresponding Utype is Observation.numSeg .

C.5.1 Description of physical axes: Characterisation classes

As mentioned in the use-cases, selection criteria for an observation depend on the physical axes contained in the dataset especially the position, band , time, and the type of observed quantity, that we call "observable" in the data model and can represent various types of flux but also velocity, etc. Such a description was tackled in the IVOA Characterisation data model from which we re-use mainly the 2 first level of details except for the spatial coverage where the support region (level 3) is used too.

C.5.1.1 Spatial axis

- The observation reference position
  Two coordinates in position are used to identify a reference position (typically the center) of an observation in the sky, attached to a coordinate system definition.
  - Coordinate system
    The coordinate system defined in the Characterisation DM is based on the STC:Coordsys class. The model in principle supports all kind of coordinate systems defined in the STC reference list. However, the Obs/Tap implementation of the model mandates that queries expressed in ICRS should be supported in an Obs/TAP service. This allows a general query to be sent to multiple archives or data centers , but requires some interpretation /conversion of coordinates at the server side. Still this is efficient for the large data discovery strategy we need to provide. Data in a specific coordinate system will be available in a second step via client applications that would do the conversions or adapt the coordinate system to some specific servers.
  - Coordinates
    The model uses the Location Class from the Characterisation DM, with the 2 Utypes values:
    Char.SpatialAxis.Coverage.Location.coord.Position2D.Value2.C1
    Char.SpatialAxis.Coverage.Location.coord.Position2D.Value2.C2
    whose short names in the ObsCore table and ICRS system are `s_ra` and `s_dec`.
- The covered region
  The Coverage class along the spatial axis provide 2 possible concepts:
  - the Bounds: a bounding box that can estimate very coarsely the coverage of an observation. It is modeled as a couple of intervals on each coordinates with Utypes:
    Char.SpatialAxis.Coverage.Bounds.limits.Interval.LoLimit2Vec.C1
    Char.SpatialAxis.Coverage.Bounds.limits.Interval.HiLimit2Vec.C1

    Char.SpatialAxis.Coverage.Bounds.limits.Interval.LoLimit2Vec.C2
    Char.SpatialAxis.Coverage.Bounds.limits.Interval.HiLimit2Vec.C2
  - the Support `s_region`
    A precise region description of spatial footprint of the dataset using region

types Circle, polygon, etc , provided in STC. Char.SpatialAxis.Coverage.Support.Area and Char.SpatialAxis.Coverage.Support.AreaType define this region, and STC-S can be used to serialise the values.

- The spatial resolution `s_resol`
  The minimal size that can be distinguished along the spatial axis, `s_resol` is stored in arcseconds and is called Char.SpatialAxis.Resolution.RefVal in the observation Utype list.
- The astrometric calibration status `astrometric_calStatus`
  A string to encode the calibration status along the spatial axis (astrometry). provide examples. Possible values could be Calibrated, Normalized, Absolute, Relative cf charac. TBD
- Astrometric precision `astrom_precision_stat`
  This parameter gives an estimate of the astrometric statistical error after the astrometric calibration phase. The corresponding Utype will be Char.SpatialAxis.Accuracy.statError.refval.
- Spatial sampling `s_pixel_scale`
  This corresponds to the sampling precision of the data along the spatial axis. It is stored as a real number corresponding to the spatial sampling period , i. e., the distance in world coordinates system units between two pixel centers. May contain two values if the pixels are rectangular.

C.5.1.2 Time axis

Three time stamps are used: `t_start` , `t_stop` , and `t_span` the elapsed time. A format like MJD is useful for easy calculations and preferred for the Observation Core components model. The elapsed time is expressed in days.

- Time resolution `t_resolution`
  This item, mapped to the Char.TimeAxis.Resolution.RefVal Utype provides an estimate or average value of the temporal resolution.
- Time Calibration Status `t_cal_status`
  This parameter gives the status of time axis calibration. This is specially useful for time series.
- Time Calibration Error `t_staterr`
  A parameter used if we can estimate a statistical error on the time measurements (for time series again).

C.5.1.3 Spectral axis

This axis is generally used to represent different kinds of physical measurements: wavelength, energy, frequency or some interpretation of this with respect to a reference position like velocity.

The data model distinguishes the various flavors of this axis using the ucd attached to it, Char.SpectralAxis.ucd or `em_ucd` . Possible values for this ucd are defined in the Spectrum DM [3] in section 4.1

- Spectral Bounds
  `em_min` and `em_max` gives the bounds of data present along the spectral axis. Such values are expressed as frequencies but using meters as units as it is easily convertible.

- Spectral Resolution `em_resol`
  A mean estimate of the resolution (FWHM of the LSF). This is can be used for narrow range spectra whereas in the majority of cases , the resolution power is preferable due to the LSF variation along the spectral axis.
- Spectral Resolution Power `em_res_power`
  The resolution power along this axis is called `em_res_power` and maps to the Char.SpectralAxis.Resolution.ResolPower Utype in the Characterisation DM.
- Resolution Power limits
  `em_res_power_min` and `em_res_power_max` parameters give the limit of variation of the resolution power in the observation.

C.5.1.4 Observable axis Most observations measure some flux quantity depending on position and/or wavelength/energy/frequency or depending on time. Here we consider a more general axis: the "Observable axis" that can be either flux or any other quantity, the nature of which is given by the UCD attached to this axis.

phot.flux.density;phys.polarization.stokes.I

The possible UCD values are part of the UCD1+ vocabulary [9]. You can find simple flux classes like : phot.flux, phot.flux.density, phot.count, phot.mag, or more complex combinations like phot.flux.density;phys.polarization.stokes.I . Table C.5.1.4 provides a list of possible triplets(observable name, ucd, units) for various observables data providers may want to describe in their archive.

Observable name	ucd for the observable (ucd1+)	description	unit
counts	phot.count	nb of photons counts	unitless
scaled counts	phot.count	recorded counts scaled using a normalisation function	unitless
ADU	phot.ADU	nb of recorded units on the detector	unitless
Flux/Luminosity			
flux	phot.flux	Photon flux	W/m2
flux density	phot.flux.density	Flux density :energy/time/area/freq interval	Jy
surface brightness	phot.flux.density.sb	Flux surface brightness	Jy/arcs2 or Jy/str
surface brightness or specific intensity	phot.flux.density.sb	Flux density per beam : energy/time/area/freq interval per solid angle	Jy/beam
surface brightness or integrated intensity	phot.flux..sb	radio continuum intensity/integrated in a pass band	Jy
flux magnitude	phot.mag	Photometric magnitude	mag
magnitude surface brightness	phot.mag.sb	Surface brightness in magnitude units	mag/arcs2
optical density	phys.density	optical density through glass plates	unitless
line ratio	phot.flux.density;arith.ratio	for line ratio maps	unitless
equivalent width	spect.line.eqWidth	for line intensities relative to continuum	nm, MHz
antenna temperature	phot.antennaTemp	Radio heterodyne obs. : Ta*, antenna Temperature	K
integrated intensity	???	radio heterodyne observations	K.km/s
flux	phot.flux	integrated flux per wl or velocity interval for lines on radio observations	Jy.km/s
Polarization (Flux)			
Stokes parameters I	phot.flux.density;phys.polarization.stokes.I		Jy.arcsec^{-2}, J beam^{-1}
Stokes parameters Q	phot.flux.density;phys.polarization.stokes.Q		
Stokes parameters U	phot.flux.density;phys.polarization.stokes.U		
Stokes parameters V	phot.flux.density;phys.polarization.stokes.V		
circular parameters RR	phot.flux.density;phys.polarization.circular.RR		
circular parameters LL	phot.flux.density;phys.polarization.circular.LL		
other params: XX YY XY YX			
Polarization Intensity	phot.flux.density;phys.polarization.linear.POLI		
Polarisation angle	phot.flux.density;phys.polarization.linear.POLA		

C.6 Provenance

Provenance contains a class to represent all the Observing configuration used to acquire an observation. Instrumental parameters are gathered here.

C.6.1 Instrument name

`instrument` This class provides the name of the instrument used for the acquisition of the observation. It is given in the model as
Provenance.ObsConfig.instrument.name and encoded as a string.

C.6.2 Access to the data

The data format as well as the url to access the data set are provided by the **Access** Class inherited from the SSA Utype list and mapped to `access_format` and `access_url` short keywords respectively. The estimated size of the data file is also given by the Size attribute and corresponds to

D Obs/TAP Schema: Examples of table definitions in SQL Data Definition Language

All TAP services that implement the ObsCore model will have the same table and column descriptions in their `TAP_SCHEMA`, with the exception of the service-specific column descriptions as described in 4. Specifically, implementations may change the description, the indexed flag, and the size (in the case of variable width columns).

Note: The following SQL was modified but not tested to match the requirements in section 4. A tested version will be included as soon as possible.

EDITOR TEMPORARY COMMENTS:Mir check/validate with existing implementation

```
delete from TAP_SCHEMA.columns where table_name = 'ivoa.ObsCore';

delete from TAP_SCHEMA.tables where table_name = 'ivoa.ObsCore';

delete from TAP_SCHEMA.schemas where schema_name = 'ivoa';

insert into TAP_SCHEMA.schemas (schema_name,description) values
( 'ivoa',
  'collection of tables for IVOA data models with standard relational mappings');

insert into TAP_SCHEMA.tables (schema_name,table_name,utype,description) values
( 'ivoa', 'ivoa.ObsCore', 'Observation', 'ObsCore 1.0' );

-- NOTE: values in the size column are implementation-specific
--       and should be set by providers
-- NOTE: values in the description may be changed/enhanced by providers
-- NOTE: all other columns have values mandated by the specification
insert into TAP_SCHEMA.columns
(table_name,column_name,datatype,unit,utype,ucd,size,principal,indexed,std,description)
values
( 'ivoa.ObsCore', 'dataproduct_type', 'adql:VARCHAR', NULL,
  'Observation.DataProductType', NULL, 128, 1,0,1,
  'type of data product (image, spectrum, timeseries, visility, eventlist, cube)' ),

( 'ivoa.ObsCore', 'calib_level', 'adql:VARCHAR', NULL,
  'Observation.calibLevel', NULL, NULL, 1,0,1,
  'calibration level (0,1,2,3)' ),

( 'ivoa.ObsCore', 'target', 'adql:VARCHAR', NULL,
  'Observation.Target.name', NULL, 128, 1,0,1,
  'name of the intended target' ),

( 'ivoa.ObsCore', 'obs_collection', 'adql:VARCHAR', NULL,
  'Observation.DataID.collection', NULL, 128, 1,0,1,
  'name of the archive or data collection' ),
( 'ivoa.ObsCore', 'obs_id', 'adql:VARCHAR', NULL,
```

```
  'TBD:Observation.DataID.creatorDID', NULL, 128, 1,0,1,
  'IVOA dataset identifier assigned by the publisher' ),
( 'ivoa.ObsCore', 'obs_publisher_did', 'adql:CLOB', NULL,
  'Observation.Curation.PublisherDID', NULL, NULL, 128, 1,0,1,
  'IVOA dataset identifier assigned by teh publisher' ),

( 'ivoa.ObsCore', 'access_url', 'adql:CLOB', NULL,
  'Observation.Access.reference', NULL, NULL, 1,0,1,
  'URL to download the data' ),
( 'ivoa.ObsCore', 'access_format', 'adql:VARCHAR', NULL,
  'TBD:Observation.Access.format', NULL, 128, 1,0,1,
  'file format or structure of downloaded product(s)' ),
( 'ivoa.ObsCore', 'access_estsize', 'adql:INTEGER', 'KB',
  'TBD:Observation.Access.size', NULL, NULL, 1,0,1,
  'estimated download size' ),

( 'ivoa.ObsCore', 's_ra', 'adql:DOUBLE', 'deg',
  'Characterisation.SpatialAxis.Coverage.Location.coord.Position2D.Value2.C1',
  NULL, NULL, 1,0,1,
  'ICRS Right Ascension of central coordinates' ),
( 'ivoa.ObsCore', 's_dec', 'adql:DOUBLE', 'deg',
  'Characterisation.SpatialAxis.Coverage.Location.coord.Position2D.Value2.C2',
  NULL, NULL, 1,0,1,
  'ICRS Declination of central coordinates' ),
( 'ivoa.ObsCore', 's_region', 'adql:REGION', 'deg',
  'Characterisation.SpatialAxis.Coverage.Support.Area',
  NULL, NULL, 1,0,1,
  'region bounded by observation' ),
( 'ivoa.ObsCore', 's_resolution', 'adql:DOUBLE', 'arcsec',
  'Characterisation.SpatialAxis.Resolution.refVal.Cresolution',
  NULL, NULL, 1,0,1,
  'typical/representative/characteristic spatial resolution' ),

( 'ivoa.ObsCore', 't_min', 'adql:DOUBLE', 'd',
  'Characterisation.TimeAxis.Coverage.Bounds.limits.Interval.StartTime',
  NULL, NULL, 1,0,1,
  'start time of observation (MJD)' ),
( 'ivoa.ObsCore', 't_max', 'adql:DOUBLE', 'd',
  'Characterisation.TimeAxis.Coverage.Bounds.limits.Interval.StopTime',
  NULL, NULL, 1,0,1,
  'end time of observation (MJD)' ),
( 'ivoa.ObsCore', 't_exptime', 'adql:DOUBLE', 'sec',
  'Characterisation.TimeAxis.Coverage.Support.Extent',
  NULL, NULL, 1,0,1,
  'exposure time of observation' ),
( 'ivoa.ObsCore', 't_resolution', 'adql:DOUBLE', 'sec',
  'Characterisation.TimeAxis.Resolution.refVal',
  NULL, NULL, 1,0,1,
  'typical/representative/characteristic Time resolution' ),
```

```
( 'ivoa.ObsCore', 'em_min', 'adql:DOUBLE', 'm',
  'Characterisation.SpectralAxis.Coverage.Bounds.limits.Interval.LoLim',
  NULL, NULL, 1,0,1,
  'minimum wavelength' ),
( 'ivoa.ObsCore', 'em_max', 'adql:DOUBLE', 'd',
  'Characterisation.SpectralAxis.Coverage.Bounds.limits.Interval.HiLim',
  NULL, NULL, 1,0,1,
  'maxmimum wavelength' ),
( 'ivoa.ObsCore', 'em_res_power', 'adql:DOUBLE', NULL,
  'Characterisation.SpectralAxis.Resolution.ResPower.refVal',
  NULL, NULL, 1,0,1,
  'typical/representative/charateristic spectral resolution' ),

( 'ivoa.ObsCore', 'o_fluxucd', 'adql:VARCHAR', NULL,
  'Characterisation.ObservableAxis.ucd', NULL, 32, 1,0,1,
  'UCD describing the observable axis (pixel values)' )
;
```

E OBS/TAP Implementations

E.1 OBS/TAP implementation at CADC

tbc

E.2 OBS/TAP service for XMM SSC catalog Data in the framework of Saada

TBC, Laurent

CDS and XMM/SSC catalog team implemented a version of the Obs/TAP service on the XCatDB database, one of the public interfaces to the XMM-Newton catalogue [5] The XCatDB contains millions of X-ray data files (time series, spectra, images, catalogs) distributed in various formats (FITS, HTML, PNG, PDF, ZIPPED folders). An extension to the Saada application [6], supporting TAP access and using the Observation data model core components, allows managing this heterogeneous data collection.

Fig. 4 shows the workflow launched for a query on the service developped at CDS by G.Mantelet.

This implementation is based on a smart ADQL editor developed on high level Javascript features (Drag and Drop, Ajax). The query responses are generated by default as VOTable XML documents displayed, via XSLT transformation, in a user-friendly panel, gathering lists of active urls to point to each individual result. Fig.5 shows how a user can submit a query with this interface, and Fig.6 exposes the interface to the query response. More output formats for the query response are available : HTML, CSV,TSV. This implementation takes benefits of extended access formats for data products. For instance, making spectrum plots available in visual format(pdf) allows end-users to have a quick look before downloading larger bulks of data. Spectra are delivered in ZIP balls containing all data files relative to an observation and the attached calibration/instrumental files needed

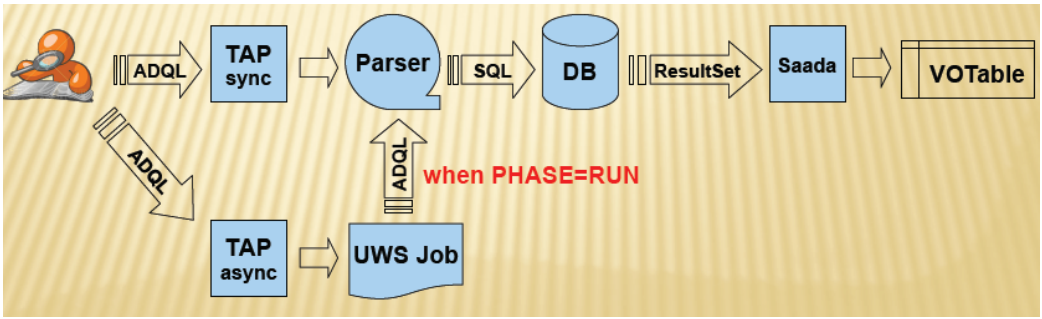

Figure 4: The workflow executed by the SSC XMM Obs/Tap service for resolving a query. It has two execution modes as required for TAP: a synchronous one , top branch and an asynchronous one, using the UWS protocol.

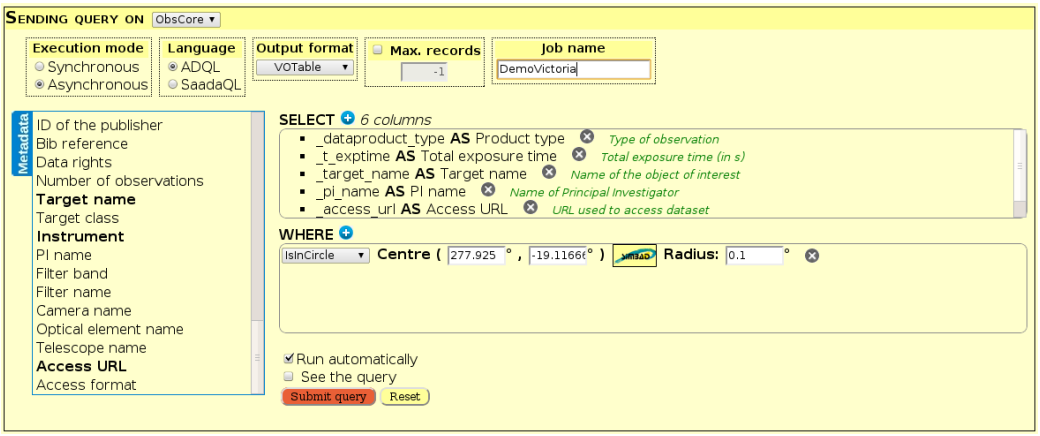

Figure 5: User interface to the XMM SSC catalog Obs/TAP service in Strasbourg, with a specified query

to achieve a complete calibration. This service remains very relevant although only a few number (24?) of the ObsCore table fields can be populated due to the X-ray data specificity.

It will be open with the XCatDB release, and the TAP service will be included in the Saada release. Both Java ADQL parser and UWS libraries will be made available as public tools on the Saada Web site. [6]

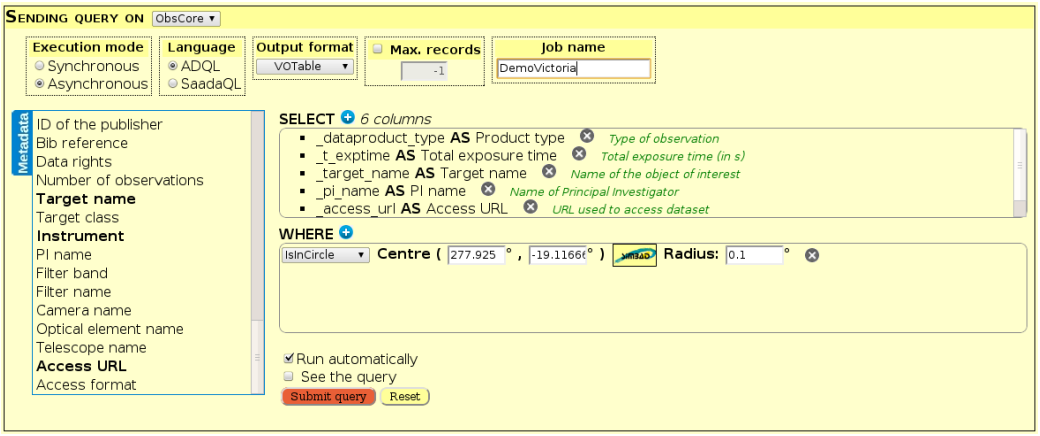

Figure 6: An example of a query response from the XMM SSC catalog Obs/TAP service to the query submitted in Fig.fig:xmmui

F Updates of the document

- version 0.2 to 1.0
  - include implementation part in section 3
  - fix underscore character in most places
  - include data model summary table for all fields in appendix A
  - add a status column for each field M or R or O
  - update tables `em_domain` moved up (minimal change)
- version 1.0 to 1.1
  - section 5: re-write XMM SSC Obs/TAP service description
  - introduce use case shortly at beginning and point to appendix
  - moved data model summary table back to data model section