



Universidad de Granada

Bases de Datos Multimedia para Radioastronomía: RADAMS y DSS-63

Memoria de Investigación para la obtención del
Diploma de Estudios Avanzados en el programa de
doctorado Tecnologías Multimedia

Juan de Dios Santander Vela,
Instituto de Astrofísica de Andalucía (CSIC)

Tutor: **Juan Manuel López Soler (UGR)**
Directores: **Lourdes Verdes-Montenegro Atalaya (IAA-CSIC),**
José Francisco Gómez Rivero (IAA-CSIC)

Septiembre, 2006

Índice general

Índice de figuras	III
Índice de cuadros	IV
Agradecimientos	IX
Resumen	1
Abstract	3
1 Introducción	5
1.1. El Observatorio Virtual (Virtual Observatory, VO)	6
1.2. VO para la Radioastronomía	8
1.3. Las tecnologías multimedia en el ámbito del VO	8
1.4. Estructura de esta memoria	10
2 Data Models: Definition and Properties	11
3 Goals and properties of the DSS63 Robledo Archive Data Model (RADAMS)	13
4 Existing Work	17
4.1. Data Model for Observation	17
4.2. Data Model for Astronomical Dataset Characterisation	17
4.3. IVOA Spectral Data Model	18
4.4. IVOA Data Model for Raw Radio Telescope Data	18
4.5. Other references	18
5 Overview of the DSS63 antenna	21
5.1. Spectral observations with the DSS63 antenna	22
6 RADAMS High Level Description	27
6.1. Observation	27
6.2. ObsData	28
6.3. Characterisation	28

6.4. Provenance	29
6.5. Target/Field	30
6.6. Packaging	30
6.7. Policy	30
6.8. Curation	30
7 Detailed description of RADAMS classes	31
7.1. An scientific case: water masers' survey within Bok globules . .	31
7.2. ObsData and Characterisation	32
7.3. Provenance	51
7.4. Target	58
7.5. Packaging	62
7.6. Policy	62
7.7. Curation	65
8 Conclusiones y trabajo futuro	69
A Archive Structure and Implementation	71
A.1. Instrument Control System	72
A.2. Archive Backend	72
A.3. Archive Services	73
A.4. Interfaces	73
B Archive Workflow	75
B.1. File selection	75
B.2. Automatic metadata generation	76
B.3. Web-based interface for metadata edition	77
B.4. Data query and XML and VOTable serialization	77
C Policy Determination	79
D VOPack	81
Bibliography	85

Índice de figuras

2.1. Sample Class and Attributes diagram	12
5.1. DSS63 70-meter antenna	25
6.1. RADAMS general class organization	28
7.1. ObsData class data model	33
7.2. Spatial axis frame metadata	34
7.3. Temporal axis metadata	39
7.4. Spectral axis metadata	43
7.5. Observable axis metadata	48
7.6. Provenance.Instrument data model	52
7.7. Provenance.AmbientConditions data model	58
7.8. Provenance.Processing data model	59
7.9. Target data model	61
7.10. Policy data model	63
7.11. Curation data model	67
A.1. High level, layered architecture for the Robledo Archive	71
C.1. Role determination algorithm	80
D.1. VOPack structure	82
D.2. VOPack schema listing	83

Índice de cuadros

5.1. DSS63 antenna, receiver and spectrometer properties	24
5.2. DSS63 properties, versus other antennas	25
7.1. AxisFrame.Spatial metadata	36
7.2. Coverage.Spatial.Location metadata	37
7.3. Coverage.Spatial.Bounds metadata	37
7.4. Coverage.Spatial.Support metadata	37
7.5. Coverage.Spatial.Sensitivity metadata	38
7.6. Coverage.Spatial.Resolution metadata	38
7.7. Accuracy.Spatial metadata	38
7.8. AxisFrame.Temporal metadata	40
7.9. Coverage.Temporal.Location metadata	40
7.10. Coverage.Temporal.Bounds metadata	41
7.11. Coverage.Temporal.Support metadata	41
7.12. Coverage.Temporal.Resolution metadata	41
7.13. Accuracy.Temporal metadata	42
7.14. AxisFrame.Spectral metadata	44
7.15. Coverage.Spectral.Location metadata	45
7.16. Coverage.Spectral.Bounds metadata	45
7.17. Coverage.Spectral.Support metadata	45
7.18. Coverage.Spectral.Sensitivity metadata	46
7.19. Coverage.Spectral.Resolution metadata	46
7.20. SamplingPrecision.Spectral metadata	46
7.21. Accuracy.Spectral metadata	47
7.22. AxisFrame.Observable metadata	49
7.23. Coverage.Observable.Location metadata	49
7.24. Coverage.Observable.Bounds metadata	50
7.25. Coverage.Observable.Support metadata	50
7.26. Coverage.Observable.Resolution metadata	50
7.27. SamplingPrecision.Observable metadata	50
7.28. Accuracy.Observable metadata	51
7.29. Provenance instrument metadata	53
7.30. Instrument location metadata	53

7.31. Antenna configuration metadata	54
7.32. Feed configuration metadata	54
7.33. Beam configuration metadata	55
7.34. Receiver metadata	55
7.35. Spectrum metadata	56
7.36. Velocity metadata	57
7.37. AmbientConditions metadata	58
7.38. Opacity metadata	59
7.39. Processing Step	60
7.40. Calibration metadata	61
7.41. Policy metadata	64
7.42. Policy related Users metadata	64
7.43. Policy related Project metadata	65
7.44. Policy related DataID metadata	66

Desde que orbitaron los primeros satélites, hacía unos cincuenta años, billones y cuatrillones de impulsos de información habían estado llegando del espacio, para ser almacenados para el día en que pudieran contribuir al avance del conocimiento. Sólo una minúscula fracción de esa materia prima sería tratada; pero no había manera de decir qué observación podría desear consultar algún científico, dentro de diez, o de cincuenta, o de cien años. [...] Formaban parte del auténtico tesoro de la Humanidad, más valioso que todo el oro encerrado inútilmente en los sótanos de los bancos.

Arthur C. Clarke, 2001: *Una Odisea Espacial* (1968).

Agradecimientos

En primer lugar, gracias Lourdes por haber apostado por un *veterano* para tu proyecto, haberme dado confianza para que este trabajo sea posible, y por todas tus lecciones sobre el trabajo en equipo. Gracias también a Emilio García por sus comentarios, su trabajo, y su amistad. A Stephane Leon por sus cafés con churros para desentrañar mis diagramas. A Dani “i-Dan” Espada, Pepe Sabater, Gilles Bergond y Simon Verley por las charlas acompañadas de cerveza. A todos los compañeros del Instituto de Astrofísica de Andalucía con los que también he compartido cafés, té, desayunos, comidas, risas, y lances futbolísticos. Mención especial a José Francisco Gómez por haberme guiado a través de todos los detalles de DSS63, y sus siempre valiosos comentarios sobre el modelo. Tampoco quiero olvidarme de los compañeros del LAEFF-INTA y de IRAM que me ayudaron con el contenido.

Y por último, gracias a mis padres por haber tenido paciencia con mis vueltas y revueltas hasta llegar aquí.

Resumen

Todo el esfuerzo que la comunidad astrofísica ha invertido en el Observatorio Virtual (Virtual Observatory, VO) ha sobrepasado el punto de no retorno: el VO es una realidad hoy, y es una iniciativa que se auto sustenta, y a la que deben adherirse todos los proyectos de archivo. Aquí mostramos cómo se pueden utilizar todos los modelos de datos propuestos para el VO para la creación de un modelo de datos para un archivo radioastronómico multi-instrumento de antena única (RADAMS), habiendo seleccionado para dicho trabajo la antena DSS63 de 70m, en la estación DSN de NASA en Robledo de Chavela. Hemos comenzado a diseñar su archivo científico, y proponemos una exhaustiva lista de Descriptores Universales de Contenido (Universal Content Descriptors, UCDs), y palabras clave FITS para todos los metadatos de interés. Este modelo de datos será refinado en una etapa posterior, en base a la experiencia que obtengamos con su implementación. Veremos también cómo, si consideramos los archivos astrofísicos como un caso particular de archivo multimedia, podremos utilizar estándares multimedia orientados al archivo como MPEG-7 y MPEG-21 como base para los metadatos sobre Provenance, Policy, Packaging y otros.

Abstract

All the effort that the astrophysical community has put into the development of the Virtual Observatory (VO) has surpassed the non-return point: the VO is a reality today, and an initiative that will self-sustain, and to which all archival projects must adhere. Here we show how we can use all VO proposed data models to build a VO-compliant single-dish, multiple-feed, radio astronomical archive data model (RADAMS). For this task we have selected the DSS63 70-m antenna at NASA's DSN station in Robledo de Chavela. We have started the design of its scientific archive, and we propose an exhaustive list of Universal Content Descriptors (UCDs) and FITS keywords for all relevant metadata. We will further refine this data model with the experience we gain from that implementation. We will also show how if astrophysical archives are considered as a special form of multimedia archives, we will be able to use archive-oriented multimedia standards such as MPEG-7 and MPEG-21 as a basis for Provenance, Policy and Packaging metadata.

Capítulo 1

Introducción

El proyecto AMIGA pretende estudiar una muestra de galaxias aisladas compuesta por más de 1000 objetos mediante el uso de datos multifrecuencia, con un énfasis especial en datos radio en longitudes de onda centimétricas, milimétricas y submilimétricas. Toda esta información se publica periódicamente a través de la página web del proyecto:

`<http://www.iaa.csic.es/AMIGA.html>`

Esta web pronto contará con un interfaz de Observatorio Virtual para permitir la obtención de información a la carta.

Como extensión natural del proyecto AMIGA surgió AMIGA+, que ha obtenido recientemente financiación para tres años (del 1 de Enero de 2006 al 31 de Diciembre de 2008), y que entre sus metas incluye la aplicación de la experiencia obtenida con AMIGA en el ámbito de las bases de datos y la radioastronomía, participando en el desarrollo de software y archivos para observatorios astronómicos milimétricos y submilimétricos, tanto de antena única como interferométricos.

Los dos observatorios interferométricos considerados son SMA (Sub-Millimeter Array) y ALMA (Atacama Large Millimeter Array). SMA es el primer interferómetro submilimétrico actualmente en operación, y precursor del extremadamente complejo proyecto ALMA (8 antenas para SMA frente a 50-64 antenas para ALMA).

En cuanto a radioastronomía de antena única, hemos comenzado a colaborar con la Deep Space Station de NASA en Robledo de Chavela, para crear un archivo de las observaciones espectroscópicas realizadas por la antena de 70m DSS63. Esta actividad forma parte de la colaboración conjunta IAA-LAEFF dentro del marco de la Red Temática Observatorio Virtual Español, financiada por el Plan Nacional de Astronomía y Astrofísica, AYA2005-24102-E.

La meta de del trabajo presentado en esta memoria, por tanto, es proporcionar una versión inicial de un modelo de datos para un archivo radioas-

trómico multi-instrumento apropiado para la antena DSS63, que sea lo más general posible para poder adaptarlo a otras antenas.

1.1. El Observatorio Virtual (Virtual Observatory, VO)

El modelo de datos que se propone se basa en los trabajos de modelado de datos ya existentes dentro del ámbito del VO. Por lo tanto, es imprescindible realizar una introducción al VO en este punto.

El campo de la Astrofísica ha utilizado siempre gran número de recursos de computación y de almacenamiento de datos. Miles de millones de bytes de datos se generan cada noche en todos y cada uno de los observatorios, como resultado de las campañas de observación realizadas por múltiples astrónomos. El mismo grupo de investigación que realizó la propuesta de observación normalmente dispone de los datos procesados, y los datos reales no suelen estar disponibles para el resto de los astrofísicos.

Hoy en día, los instrumentos proporcionan los datos directamente en forma digital, y la resolución ha mejorado de forma exponencial. Masivos proyectos de observación del cielo, como el Sloan Digital Sky Survey¹, proporcionan terabytes de datos a la comunidad, así como datos procesados, de modo que se puede hacer ciencia con ellos de forma más sencilla. El problema principal, entonces, es cómo poner la enorme cantidad de datos proporcionada por cada observatorio a disposición de todos los astrónomos, permitiendo que sea fácil encontrar esos conjuntos de datos. Este problema se agrava con el creciente tamaño de esos conjuntos de datos, debido al ya mencionado incremento en resolución y sensibilidad.

La Astronomía ha contado durante años con un formato común de datos para el intercambio de datos astrofísicos: el Flexible Image Transport System (FITS). Este formato de datos fue inventado a finales de los 70, y es capaz de almacenar un número arbitrario de cabeceras y unidades de información (Headers and Data Units, HDUs), cada uno de los cuales consistente en una cabecera de texto que describe una unidad de datos binaria de tamaño arbitrario, almacenada de forma binaria, con el formato descrito por la cabecera.

Sin embargo, un formato común de datos no lo es todo. Diferentes observatorios utilizan diferentes palabras clave en sus cabeceras, y el mismo tipo de información se almacena en HDUs diferentes en cada observatorio.

Por consiguiente, la solución a estos problemas debe cumplir las siguientes restricciones:

- Debe existir un mecanismo para encontrar cada uno de los diferentes repositorios de datos que estén disponibles en cada momento.

¹<http://www.sdss.org/>

1.1. EL OBSERVATORIO VIRTUAL (VIRTUAL OBSERVATORY, VO) 7

- Los diferentes repositorios de datos deben estar descritos y caracterizados, de modo que se pueda tener en cuenta sólo aquellos que contengan información que cumpla una serie de criterios que fijará cada astrónomo.
- La descripción de los datos estará unificada, de modo que será más fácil operar con datos provenientes de fuentes heterogéneas.
- Transferencia de datos minimizada entre archivo y usuario; el procesamiento de los datos se realizará preferentemente en el servidor, y se devolverán los resultados al astrónomo.

El VO, por tanto, es un conjunto de prácticas y modelos de datos que facilitan el descubrimiento de repositorios de información interoperables, cuya descripción se ha unificado por medio de un modelo de datos común.

En la actualidad, se pueden realizar algunas actividades de procesamiento sobre los datos en el servidor, pero para muchas operaciones aún es necesario que el usuario obtenga un conjunto de datos en bruto para su procesamiento. Sin embargo, el usuario puede contar con un formato de datos común, y una descripción exhaustiva de los datos obtenidos del VO. En una etapa posterior, todos los análisis se realizarán de forma remota sobre los datos.

Obviamente, las propiedades anteriormente mencionadas para los archivos de información sólo pueden lograrse si existe una entidad para la creación de estándares. Tal entidad, en el caso del VO, es la Alianza Internacional para el Observatorio Virtual (International Virtual Observatory Alliance, de aquí en adelante IVOA). IVOA es una federación de grupos VO nacionales y supranacionales, y dirige y sanciona el desarrollo de los diferentes componentes de la infraestructura VO.

En este sentido, es útil pensar en el VO como en otra clase de instrumento astrofísico, que permite obtener información multifrecuencia de diferentes observatorios al mismo tiempo.

Pero el VO puede hacer por la ciencia mucho más que proporcionar al astrónomo toda la información disponible sobre una determinada parte del cielo. Las herramientas actuales del VO permiten que los científicos creen flujos de trabajo que descubran tipos particulares de objetos —por ejemplo, utilizando técnicas de diagnóstico para proporcionar objetos candidatos a pertenecer a una determinada clase de objeto—, o incluso más interesante: el VO permitirá el uso de técnicas de minería de datos, junto al modelado de los mismos, para realizar la clasificación automática o asistida de objetos que pueda proporcionarnos, por ejemplo, una lista de objetos exóticos, que no se ajusten a ninguna categoría preestablecida.

Estos instrumentos virtuales del VO necesitan que sus observaciones puedan llegar a su público. Ése es el papel de los protocolos de IVOA. Existen varios de ellos, pero los más importantes para nuestro trabajo son:

- **ConeSearch:** permite consultas sobre un par de coordenadas RA/Dec (Ascensión recta/Declinación), junto a un radio de búsqueda (con lo que

se define un cono en el cielo), y devuelve apuntadores a objetos o entradas de catálogo que pertenezcan a ese cono en concreto. Se considera un protocolo obsoleto, que se mantiene por compatibilidad.

- **SSAP (Simple Spectral Access Protocol):** permite consultas sobre regiones complejas del cielo (ConeSearch es un subconjunto de este protocolo), y también la especificación de rangos concretos de frecuencias. Los objetos devueltos siempre apuntan a datos espectrales o de series temporales.
- **SkyNode:** es el protocolo más alto en la jerarquía, y permite interrogar directamente al servidor, proporcionando infraestructura para la coincidencia cruzada (cross-matching) con fuentes de otros catálogos.

1.2. VO para la Radioastronomía

Una de las ventanas más interesantes del espectro electromagnético para la Astronomía, que complementa muy bien a la parte visible, del mismo, es la ventana radio, dado que estas longitudes de onda se ven mucho menos afectadas por fenómenos de extinción.

Aun así, la Radioastronomía es relativamente nueva, especialmente cuando se compara con la Astronomía óptica, y los no radioastrónomos a menudo no están familiarizados con las técnicas especializadas de esta disciplina. De aquí la importancia de una infraestructura VO para la radioastronomía, que proporcione un acceso sencillo a herramientas y datos.

Realizando una búsqueda de archivos radio que cumplieran estándares VO, sólo encontramos el ATCA (Australian Telescope Compact Array) Data Model [1], y el NOAO Science Archive Domain Model [2], pero incluso aunque proporcionan interfaces VO, hacen uso de muy pocos modelos estándar, y en lugar de ello se basan en sus propias implementaciones. La principal contribución de este trabajo, por tanto, es la selección entre todos los estándares VO propuestos de los adecuados para configurar el archivo científico para DSS63, la descripción y ampliación detallada de los mismos, y su uso intensivo para describir, almacenar y recuperar observaciones espectroscópicas y de continuo en múltiples bandas, con datos multi-instrumento.

Además, este trabajo constituye la base sobre la que se implementará el archivo de la antena DSS63 de 70m, y contribuye al desarrollo del VO en radio.

1.3. Las tecnologías multimedia en el ámbito del VO

Las señales astrofísicas son un caso particular de señal multimedia. Una observación astronómica puede proporcionar señales unidimensionales (es-

pectros, o variaciones temporales de luminosidad, por ejemplo), señales bidimensionales (imágenes ópticas, mapas de radio...), o incluso señales de tres o más dimensiones (cubos de datos de interferometría, series temporales de imágenes, series temporales de cubos de datos...).

Dentro del entorno VO se impone el estudio simultáneo de la información de múltiples instrumentos, para lo cual es necesario realizar muy diferentes operaciones de procesado de señal, en muchas ocasiones de forma conjunta, sobre los diferentes archivos multimedia utilizados. Las técnicas de minería de datos que se utilizarán en este entorno están basadas en las técnicas habituales de modelado, y de aplicación de conocimiento a priori.

Como veremos a lo largo de la memoria, dentro de la infraestructura del VO existen dos niveles de descripción imprescindibles para la explotación de un archivo, y se corresponden casi directamente con dos estándares establecidos dentro de la transmisión de contenidos multimedia en Internet. El primer nivel de descripción es la caracterización (*Characterisation* en inglés) de los datos: qué contienen realmente, cual es la resolución, qué parte del espectro, del tiempo, del espacio, estamos cubriendo, cuál es la sensibilidad del instrumento, y cuáles son sus zonas de muestreo... Parte de esos metadatos (resolución, tamaño...) se incluyen dentro de la mayoría de los formatos multimedia, pero la información sobre formas, movimientos, y localización, junto con la identificación de puntos de interés concretos, son la razón de ser del estándar MPEG-7 [3].

El segundo nivel de descripción afecta a la forma en que se entregan los datos, y los derechos que pueden tener diferentes usuarios sobre esos diferentes datos. En el marco de VO, hablaremos de *Packaging* y *Policies* (véanse más adelante las secciones 7.5 y 7.6), mientras que en el marco multimedia estaríamos hablando de Recursos, Contenedores, Contenidos, y Derechos de Uso. El estándar correspondiente para el VO aún está por definir, y a lo largo de esta memoria veremos cómo el estándar MPEG-21 [4] (que abarca conceptos como Elementos Digitales, Derechos de Uso o Gestión y Protección de Propiedad Intelectual, entre otros) nos inspira en la definición del modelo de empaquetado de datos, y las políticas de acceso a los mismo.

Las herramientas de etiquetado multimedia basadas en MPEG-7 no han vivido un éxito comparable al de otros estándares del Motion Pictures Expert Group (MPEG), fundamentalmente debido a la falta de automatización de las mismas, y la continua necesidad de supervisión especializada para conseguir una descripción completa de las diferentes escenas que conformarían un contenido multimedia. Afortunadamente, el mayor conocimiento a priori disponible sobre las señales astrofísicas permite reducir al mínimo la necesidad de intervención humana, permitiendo además la elección de valores por defecto útiles para la inmensa mayoría de los casos.

1.4. Estructura de esta memoria

Esta memoria comienza con un amplio repaso al concepto de Modelo de Datos, y por qué es necesario tanto por razones científicas como relacionadas con el VO, en el capítulo 2. Los objetivos detallados del archivo se explican en el Capítulo 3, con un breve repaso al trabajo existente en este campo en el Capítulo 4. Terminamos con el material introductorio con un repaso a las capacidades y modos de observación de la antena DSS63 de 70m en el Capítulo 5. Con ese bagaje, realizamos un repaso de alto nivel del sistema de archivo en el Capítulo 6, y refinamos el detalle de los diferentes componentes del archivo en el Capítulo 7, teniendo especial relevancia las Secciones 7.2, 7.5 y 7.6. La memoria termina en el Capítulo 8, en el que presentamos las conclusiones y el trabajo futuro.

Proporcionamos diversos apéndices con información adicional: proponemos un diseño para infraestructura de archivo que se implementará en el Apéndice A, y para el flujo de trabajo del archivo en el Apéndice B. Especificamos el algoritmo de determinación de la política de acceso en función de los usuarios en el Apéndice C, y el concepto de VOPack en el Apéndice D.

Todos los capítulos y apéndices se han escrito en inglés para permitir un rápido aprovechamiento de los mismos, y facilitar la colaboración de la comunidad de IVOA para el establecimiento de los estándares propuestos para el VO. El capítulo de Conclusiones y Trabajo futuro también se presenta en castellano.

Durante mi colaboración con el grupo AMIGA he participado en diferentes comunicaciones y artículos, que se detallan en una segunda memoria sobre los cursos de doctorado, y publicaciones realizadas.

Chapter 2

Data Models: Definition and Properties

A data model is a description of the set of entities needed for information storage in a particular field, and specifies both the data being stored, and the relationships between them.

Applying this definition for a single-dish, multiple-feed Radio Astronomy instrument, the data model is the abstract, logical view for describing, tagging and grouping all possible information that defines the different observation modes available in the target instrument, Robledo de Chavela's 70-m DSS63 antenna. This description will try to be as general as possible, in order to minimise the needed modifications to accommodate it to additional observing modes or stored data products.

Hence, the goal of this data model is double. First, it has to serve as the blueprint for the internal organization of the information that will be stored in the archive: what kind of information will be stored in the database, how will it be grouped, and what relationships will be implemented. Therefore, the mapping between the data model and the set of tables, fields and indices on the database will be obtained.

Second, it will provide us with all the information to be included in the headers of the final products for an observation, either in FITS or VOTable format, in order for those headers to be completely standard in the VO sense.

A data model is made up of Classes (blueprints for information containers) and Attributes (metadata) describing those classes, as well as the relationships between them. A particular piece of data will be represented by an Instance of a Class, and will have particular values for its Attributes. Relationships are established through the Attributes in common between Classes (indices). In particular, we will group all Instances that share the same value for a common Attribute.

When creating a new data model in the VO environment, compliance with current standards, both for data description and archive organization, allows

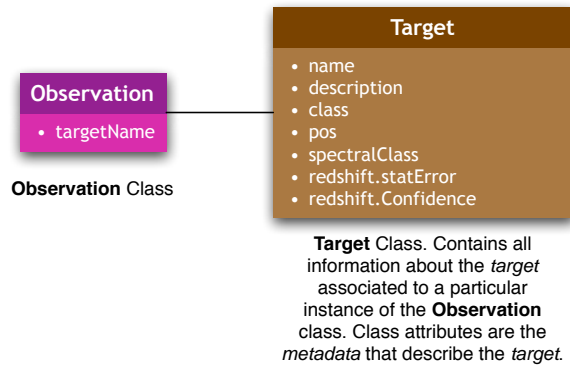


Figure 2.1: Sample Class and Attributes diagram

us to ensure:

- Interoperability for the interchange of information (by means of common metadata).
- Access and Query services standardization (both for man-machine and machine-machine interactions).
- Easy automated visualization, processing and analysis in the VO environment.

Chapter 3

Goals and properties of the DSS63 Robledo Archive Data Model (RADAMS)

A successful implementation for the RADAMS needs to:

- Reproduce the scientific domain for any radio observation performed with the DSS63 antenna, by means of proper instrument and data characterisation.
- Be as general and flexible as possible, so that new capabilities can be easily added.
- Support a set of representative queries.
- Follow IVOA directives, in order to be VO standards compliant, and also to help in the archive's flexibility.

The model will hold information about the following properties:

- Datasets.
- Instrumental and observation configuration.
- Observational conditions, fundamentally weather conditions.
- Observation projects, proposals, etc.
- Calibration status and data processing history.
- Numerical representation for FITS/VOTable serializations.
- Curation data (i.e., who is providing the datasets) for inclusion in VO registries.

Those characterisations will allow for many different queries. The queries that one will be able to perform will be, at least, queries by:

- Object/Target/Source.
- Field (through standard IVOA protocols, such as the SkyNode protocol, or the ConeSearch protocol).
- Observation date.
- Frequency Range/Spectral band.
- Spectral characteristics.
- Calibration type.
- Instrumental settings.
- Location (in coordinates, velocities, etcetera).
- Atmospheric conditions.
- Data quality (using numeric or descriptive qualifiers).
- Project management data, such as PI, Observer, etc.

These queries require the archive to provide an interface implementing, at least, the following VO services:

- SkyNode.
- ConeSearch.
- SSAP (for spectral access).
- SIAP (eventually, for map imaging).

Any agent (human or machine) interacting with the archive needs to retrieve both the data themselves, and the following information on the observation:

- Source pointing.
- Weather conditions.
- Calibration.

In addition, the system will provide, at a later stage, means to retrieve additional reports accumulated during a given term, such as:

- Weather conditions for such term.
- Opacity statistics.
- Skydips and Opacity curves.
- Engineering data.

Chapter 4

Existing Work

In order to make the RADAMS as VO compliant as possible, we have based it in the following references from IVOA:

4.1. Data Model for Observation

The Data Model for Observation (DMO, version 0.23, IVOA Data Model WG Internal Draft) [5] describes a data model for the concept of an astronomical observation, considering it as a VO entity. It describes a class that is qualified by three more classes:

- **Characterisation:** How to describe the data to be archived, in terms of data properties.
- **Provenance:** How the data were originated (instrumental configuration, observing conditions, pipelines, etc).
- **Curation:** Extra information needed for systematic resource description for inclusion in VO registries.

The DMO does not describe in detail any of these additional qualifying classes, and leaves it up to further IVOA documents; for now, only Characterisation has been specified.

4.2. Data Model for Astronomical Dataset Characterisation

This document (DMAC, version 0.9, IVOA Data Model WG Note) [6] details the Characterisation part of the DMO, that is, how any kind of observation is described in the VO.

In particular, it recommends that any archive should provide different Axes for any piece of observational data:

- Location (at least, 2D spatial coordinates).
- Temporal (time, and time sampling).
- Spectral.
- Observable quantity or quantities, such as flux, polarization, etc.

For each of these axes, data resolution and precision have to be included, as well as the associated accuracy.

One of the most remarkable aspects of the Characterisation data model is that different levels of detail can be provided in the description of the data, depending on the intended VO use of the archive: if only discovery and query services are to be provided, the level of needed detail is lower than necessary for VO data mining applications, and the archive does not have to provide characterisation for the sensitivity in different axes.

However, we will describe this archive at the highest level of detail, so that in principle the RADAMS will support data mining capabilities.

4.3. IVOA Spectral Data Model

The SDM (SDM, version 0.95a, IVOA Data Model WG Working Draft) [7] defines a particular data model for defining both spectra and time series, as particular cases of a Spectral Energy Distribution (SED) object. We will use the SED data model to represent spectroscopic data for VO web services such as the Simple Spectral Access Protocol.

4.4. IVOA Data Model for Raw Radio Telescope Data

This document (RDM, version 0.1, IVOA Radio Astronomy Interest Group Note) [8] is a draft for a data model proposal for radio-interferometer data. Single-dish instruments are treated as a particular case of a single antenna interferometer.

Lamb and Power's RDM constitutes the only IVOA specific document for radio astronomy. Many aspects of our work are based upon this model, which we have updated with the latest IVOA discussions about data models for observations, characterisation, etceteras, and we have adapted it to single-dish observations such as those performed at Robledo.

4.5. Other references

We have also made use of several other references that can be found on the Bibliography section of this memory. An special mention is made of the

MPEG-7 and MPEG-21 overview documents [4, 3], issued as web-pages hosted by <http://www.chiariglione.org/>.

Chapter 5

Overview of the DSS63 antenna

As we have stated before, the main goal for this memory is to describe the RADAMS as a blueprint for the DSS63 archive. Therefore, an introduction to the DSS63 antenna, and its observation modes and properties is mandatory. This introduction is partially based upon the second chapter of de Gregorio-Mosalvo's Ph.D. thesis [9].

The DSS63 is one of the antennas at the Madrid Deep Space Communications Complex. Three Deep Space Communications Complexes (DSCCs) were created by NASA in the late 50's, and were located at Canberra (Australia), Madrid (Spain) and Goldstone (USA), in order to allow for continuous monitoring of the incoming data from Earth-orbiting and interplanetary spacecraft missions, as well as radio and radar astronomy observations for the exploration of the Solar System and the Universe. The combined operation of the three DSCCs is what it is known as the Deep Space Network (DSN), which is managed by the Jet Propulsion Laboratory (JPL).

Of all the time devoted for astronomical observations, around 3% of the time at the Canberra and Madrid stations (up to 260 hours per antenna) is available to Host-Country astronomers. The organization responsible for the scheduling of this time at Madrid DSCC is the Laboratorio de Astrofísica Espacial y Física Fundamental (LAEFF) of the Instituto Nacional de Técnica Aeroespacial (INTA), by arrangement with NASA.

Each DSCC has at least four operational antennas:

- One 26-meter diameter antenna, originally built to support the Apollo missions to the Moon, presently used for communicating with Earth-orbiting spacecrafts.
- One 34-meter diameter high efficiency antenna (HEF), designed around a precision-shaped reflector, for maximum signal sensitivity.

- One 34-meter diameter beam waveguide antenna (BWG), based on the HEF design, with five mirrors that reflect radio signals along a beam-waveguide tube from the antenna vertex to the equipment room, for easier maintenance access.
- One 70-meter diameter antenna, with the highest sensitivity, used for tracking the deepest space missions.

Nowadays, Host Country time at the MDSCC is dedicated to perform spectroscopic observations at K-band (i.e., wavelengths around 1 cm), with the 70-m DSS63 antenna.

Table 5.1 summarises the main properties for the DSS63 antenna, and Table 5.2 compares them with those from other similar radio telescopes. Figure 5.1 shows a picture of the 70-m antenna.

5.1. Spectral observations with the DSS63 antenna

As the main scientific use of the DSS63 antenna is the recollection of spectra, we will describe how spectroscopic observations are performed with this instrument.

The observing process for a spectrum is as follows:

- **Source selection:** first, a target source with a medium elevation at the time of observation is selected; extreme elevations introduce additional pointing errors and/or additional atmospheric effects.
- **Pointing calibrator selection:** once the source has been selected, a strong pointing calibration source near the target is chosen; minimization of antenna motion between pointing calibration and the actual observation is desirable.
- **Pointing calibration:** The antenna will be moved up and down in elevation, and clockwise and counter-clockwise in azimuth, around the expected position for the pointing calibrator. As the profile for the source conforms to a Gaussian distribution, the data can be fitted with a Gaussian, and the pointing error adjusted by comparison between the expected position of the calibrator, and the fitted maximum flux position. This correction will be applied to the coordinates where the source is expected to be.
- **Focus calibration:** The same calibrator can be used to calibrate the focus of the instrument, defined as the position of the secondary mirror that maximises the power collected by the instrument. Again, the profile for the focus, when the mirror is moved along its axis, is assumed to be Gaussian, and the fit for the maximum provides the focus.

- **On/Off source observation:** Both the source and a nearby position with no emissions have to be observed, in order to discriminate the contribution from the instrument. This is performed either by changing the position of the antenna (position switching), or by moving the secondary mirror in such a way that the main feed is focused on a different region of the sky, with no radio sources (wobbler switching). Another possibility is to compare the power of the emission from the same source at slightly different frequencies, assuming that the antenna and atmospheric noise does not change with this frequency switch (frequency switching). In the case of the DSS63, on/off observations are performed either by position switching or frequency switching.
- **Atmospheric corrections:** The amount of energy received by the instrument depends strongly on weather conditions, and on the length of the path of the signal through the atmosphere. In particular, at cm wavelengths the amount of water vapour in the atmosphere is the major contributor to atmospheric opacity (a quantity that is proportional to the probability of a photon being absorbed after travelling a given length in the atmosphere). Measurements of opacity at different elevations (tipping curves, or skydips), which correspond to different air masses (a measure of the amount of atmospheric gas in the line of sight of the instrument), are used to fit a curve that provides the atmospheric opacity. This is usually done at DSS63 once per observing session and frequency setup.

There are other corrections and calibrations to consider, but most of them can be obtained from typical values for the instrument and particular configuration, and do not contribute to illustrate the observing process with the DSS63 antenna.

Of particular interest will be parameters such as system temperature (T_{sys}), main beam solid angle (Ω_{mb}), aperture efficiency (η_{a}), and the antenna temperature scale.

The data output of the correlator is a 384-sample autocorrelation function, which by means of the Fourier transform (Discrete Fourier Transform, in this case) provides a function proportional to the power spectrum of the source. The post-processing of the observation, together with the calibration procedures, will allow us to determine the actual spectrum scale, and frequencies for the salient features of the spectrum.

Table 5.1: DSS63 antenna, receiver and spectrometer properties.

Antenna System^a	Name: Deep Space Station 63 Diameter: 70 m Type: parabolic Cassegrain Mount: azimuth/elevation Latitude: 40° 25' 52" N Longitude: 04° 14' 53" W Altitude: 865.5 m HPBW^b: 42" Aperture Efficiency (η): 49% maximum Sensitivity: 0.7 K/Jy Pointing accuracy: $\leq 10''$
	Amplifier type: cooled HEMT Frequency range: 18-26 GHz Polarization: LCP ^c or RCP ^d (default, LCP) T_{sys} (winter): 50 K T_{sys} (summer): 75 K
K-band receiver	Spaceborne-500^e Type: Digital autocorrelator BW: 2, 4, 8 and 16 MHz Num. channels: 384 Observing mode: position switching
	Spectra-Data^f Type: Fourier-transform autocorrelator BW: 1, 2.5, 5 and 10 MHz Num. channels: 256 Observing mode: frequency switching
Spectrometers	SAO4K^g Type: Digital autocorrelator BW: 400 MHz Num. channels: 4096 Observing mode: position switching

^aHPBW, aperture efficiency, sensitivity and pointing accuracy measured at 22GHz, with 40° of elevation

^bHalf-Power Beam Width

^cLeft Circular Polarisation

^dRight Circular Polarisation

^eThis is the correlator currently in used, and superseded the Spectra-Data.

^fThe Spectra-Data was the correlator initially used at the facility (from 2001 to 2003), but it is no longer operational.

^gThe SAO4K correlator belongs to the Smithsonian Astrophysical Observatory, and is used in the SAMBA survey.

Table 5.2: DSS63 properties, versus other antennas; values at 22 GHz.

Telescope	Diameter	Aperture efficiency (η_a)	Resolution	Sensitivity (K/Jy)
Effelsberg	100 m	29%	40"	0.8
GBT	100-110 m	55%	34"	1.5
Robledo DSS63	70 m	49%	42"	0.7



Figure 5.1: DSS63 70-meter antenna at Robledo de Chavela, Madrid

Chapter 6

RADAMS High Level Description

Now that we have ended with the introductory material, we will start by giving a high level overview of our data model. This model was built upon the following assumptions:

- Archived data are considered exclusively single-dish, single-feed; for multiple feeds, each feed is considered a separate entity.
- The archive will contain both single pointing continuum and spectral data; radio mapping will be implemented at a later stage.
- The archive will just provide data retrieval; it will not directly provide tools for data mining, but as we will be characterising the sensitivity, as recommended by IVOA, the archive will allow for them.

The RADAMS is built with the help of several main classes. Figure 6.1 shows the high level organization of the RADAMS classes.

We will start by defining the different classes that make up the data model.

6.1. Observation

This is the main class of the data model. It describes an arbitrarily large dataset that can be derived directly from an observation, or from several observations. Generally, we will consider the set of observations taken with the same instrumental setting, and with the same associated target.

In the particular case of the DSS63, an Observation would be made of the set of on/off observations associated with the same target, or the final, flux calibrated spectra. As we will see later, the Packaging class will allow us to have different degrees of granularity for data access.

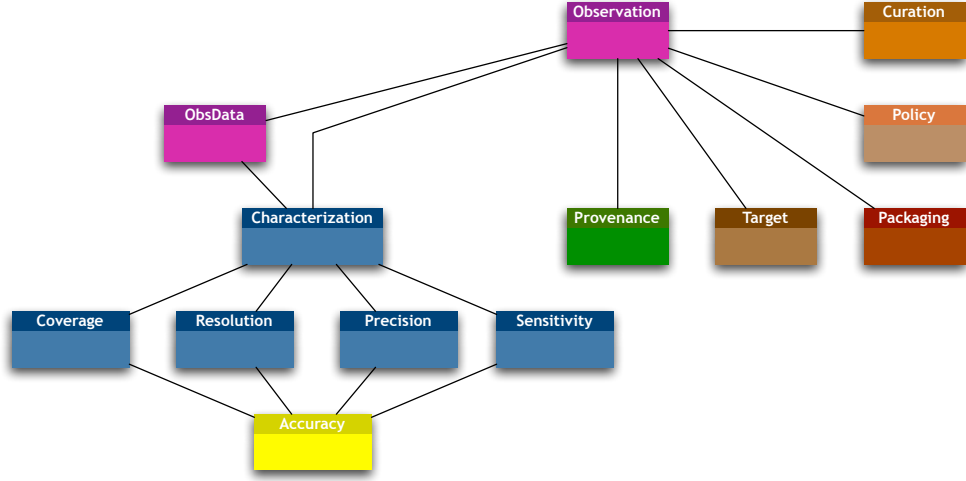


Figure 6.1: RADAMS general class organization. Different colours correspond to different sub-data models, as described by IVOA.

6.2. ObsData

This class describes the data, in tabular form. It can be just a number—if we are doing continuum observations—or an array representing a spectrum, or a set of spectra.

We propose that the ObsData for the RADAMS—that is, the final product to be stored in the archive—can be any of the following:

- Each spectrum resulting from an on/off processing, corrected by opacity and T_{sys} .
- The calibrated and averaged spectrum from all the on/off processed observations belonging to a given observation.
- Both.

The RADAMS makes use of the Packaging class—which we will define below—in order to specify what particular pieces of data are being retrieved. Besides, the archival of continuum information cannot be discarded, as most radio telescopes are capable of performing continuum scans.

6.3. Characterisation

ObsData is the main class containing scientific data, but further metadata are needed in order to describe its content quantitatively, as well as qualitatively. The class that provides this information is the Characterisation class.

The DMO document establishes that any astronomical measurement occupies a position in a multi-dimensional space, defined by several axes. Initially, this axes are just four (but more can be added): spatial (coordinates), temporal, spectral, and a fourth axis corresponding to the observed physical quantity (e.g., measured flux, or polarization, in the case of radio observations). Characterisation gives us a description of the data in different attributes and levels inside this multi-dimensional space, including physical units and scales.

The Characterisation class can be divided in the following subclasses:

- **Characterisation.Coverage:** Specifies which part of the multi-dimensional space has been encompassed by the measurement. That is, when was the observation made and for how long, which field was covered, which bands were studied, what was the range of observed flux, and so on. Each of those questions belongs to a different axis.
- **Characterisation.Resolution:** Specifies data resolution in each of the axis. Resolution is independent of the sampling, as it is a property of the instrument/observing process.
- **Characterisation.Precision:** If the data are sampled in any axis (e.g., for spectral data in the frequency domain, data are sampled), this class will include information on the sampling precision. This is different from resolution: resolution is a property of the instrument, due to uncertainties on the energy distribution of the source, because of convolution of the source's and the instrument's profiles.
- **Characterization.Sensitivity:** This class quantifies the sensitivity of the instrument in the given axis, by means of a sensitivity function that is defined for the whole coverage of the observation in the given axis. This class has not been fully defined yet by the IVOA Working Group, hence we are providing a tentative description of this class, in order to help the the efforts of the Characterisation Working Group.
- **Characterisation.Coverage.Accuracy:** For each of the above characterisations, accuracy metadata have to be provided. This class qualifies the accuracy and errors, both systematic and random, for each characterisation axis.

6.4. Provenance

This class groups all the information describing the way the dataset was created. It has to include the instrumental setting for that particular observation, coordinates for the telescope, weather and atmospheric conditions, and all the information about pre- and post- processing of data, when such data were created by processing some other.

6.5. Target/Field

This class allows the model to include relevant information about the observed target. Such target can be mapped to a known astronomical object, to a detected source, to a given field, etc. Having this entity in a separate class from the Observation one allows optimized searches by Target, and the ability to link external target information using VO services (such as Sesame, Aladin, the SkyNode interface, etcetera).

6.6. Packaging

Archive queries result in different datasets, and a particular dataset for a given Observation can contain several measurements. The Packaging class specifies what is being delivered by the archive as a result for a given query, and the organization of data packages different VOTables link to. We will provide an initial development of this class, suitable for the needs of the RADAMS. We will also propose a VO general packaging system, the VOPack.

6.7. Policy

Different observations can fall under different privacy policies. This class allows for easy tagging of private data, and for searches for public data. We will provide an initial development of this class, as needed by the RADAMS.

6.8. Curation

To be considered part of the VO community, all VO resources have to be included in a Registry. The Curation class encompasses all the metadata needed for well-formed archive and data VO registry. In addition, we also integrate an ObservingProgram subclass, proposed by Anita Richards in the DMO document, which acts as an intermediate class that allows grouping together different observations with a common goal, such as a maser survey—see Section 7.1—, for instance.

Chapter 7

Detailed description of RADAMS classes

After this initial overview of the classes that conform the RADAMS, we will detail the attributes corresponding to each of the classes of the model.

Besides, we will select appropriate FITS Keywords and UCDs for FITS and VOTable serializations of observational data. FITS Keywords will be selected first from the official NASA FITS mandatory headers [10], later from the Multi-Beam FITS Raw Data Format [11], and lastly from the NRAO GBT FITS data format [12]. Sometimes those keywords cannot appear in the main header of a FITS file, but instead in a FITS extension table. As this memory reflects the earlier stages of the archive design, **assign** means a FITS keyword has to be selected for that particular database field.

UCDs will be selected from the UCD1+ vocabulary [13], using the recommended juxtaposition technique in order to clarify the meaning of any given term. In some cases, where there is no feasible UCD combination, we will propose one.

7.1. An scientific case: water masers' survey within Bok globules

In order to better illustrate the details of the different data models, we will introduce first a sample scientific case: a water maser survey performed on Bok globules with the DSS63 [9].

Water maser emission has a rest frequency of 22235.080 MHz, and corresponds to the $6_{16} \rightarrow 5_{23}$ transition of the water molecule. This emission is excited when there is a shock wave running through the gas, which provides the energy to invert the population of the energy levels of the molecule, a prerequisite to produce a significant amount of stimulated emission. Maser emissions have very precise frequencies, and their presence is a tracer of molecular outflows or stellar mass loss.

Data collected for this survey would be part of the same Observing program, and observations performed with the same instrument and configuration would be part of the same proposal. Different observations for different targets would be tuned to different frequencies, to account for the different Doppler shifts caused by the recessions of the source.

7.2. ObsData and Characterisation

Following the ALMA Software Glossary [14], we can define several degrees of observational data:

Dump This is the smallest interval of time for which a set of correlated data can be accumulated and output from the backend stage.

Integration Set of dumps, all identical in configuration, which are accumulated and form the basic recorded unit.

Sub-scan Set of integrations performed while the antennas complete an elemental pattern across the source: e.g., frequency switching, nutator switching, etc.

Scan Set of sub-scans with a common goal, such as a pointing scan, a focus scan or an atmospheric amplitude calibration scan, among others.

Hence, the minimum useful unit to be recorded in the archive is the sub-scan, which can be appropriately described, but the minimum scientifically significant unit is the scan, so depending on the kind of granularity desired one could base the archive upon one or the other. Higher order units —such as multiple scans on the same source— should be derived from Project metadata.

To describe these scans we will use the `ObsData` class (which could also be referred to as `RawData`) as the main class for the RADAMS. From `ObsData` instances, we will form a tree of data that will describe a particular scan, both by its properties in the Spatial, Temporal, Spectral and Observable axes, and its respective Accuracy.

We show a high level overview of the `ObsData` class and its relationships with `AxisFrame`, `Coverage`, `Resolution` and `Accuracy` classes in Figure 7.1.

ObsData (also to be referred as **RawData**): Instances of this class will contain the final data. As discussed previously, we will keep either whole scans to determine the scientific data, or processed data (such as spectra).

AxisFrame: Instances of this class will contain metadata that describe the main properties of the axis, such as units, calibration state, etc.

Coverage: We will describe the position of the archived data in the parameter space: where and when was the telescope pointed, and what wavelength range was observed. Several subclasses exist:

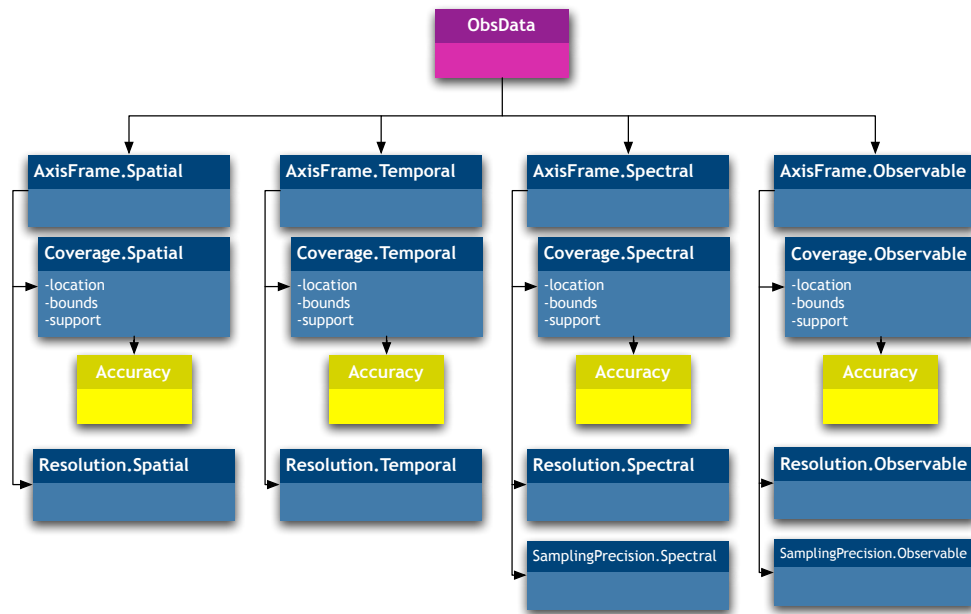


Figure 7.1: ObsData class data model; the different axes for the Characterization part of the data model are shown.

Coverage.Location: This subclass of Coverage describes the characteristic value for each axis. For instance, **Coverage.Spatial.Location** would give the central point of the observed field, and **Coverage.Temporal.Location** would hold the middle time of the scan.

Coverage.Bounds: Maximum and minimum values of the axis; for instance, **Coverage.Spectral.Bounds** would give us the maximum and minimum frequencies of the spectrum, and **Coverage.Temporal.Bounds** would give us the starting and ending time of the observation.

Coverage.Support: Set of parameters in that axis where we have valid observational data. For instance, **Coverage.Temporal.Support** could be a set of intervals when data were gathered, excluding the pauses for reissuing scans.

Coverage.Sensitivity: This can be seen as an additional refinement to **Coverage.Support**, providing a response function of the instrument in the given axis that goes beyond a Boolean response. This is especially useful for cases with a large number of small interruptions in the data, there is need for data resampling, or for which filters imperfections have to be accounted for.

Resolution: Instances of this class are used to describe resolution in each axis. For instance, `Resolution.Spatial` would give us the Half-Power Beam Width.

SamplingPrecision: For a sampled axis —such as the Spectral axis, in the case of spectroscopic data—, an instance of this class will hold sampling precision, described as a pixel scale.

Accuracy: All of the aforementioned classes should have an accompanying class in order to describe errors and data quality for each axis. In the case of archives without data mining capabilities, we only have to provide `Accuracy` instances related to the `Coverage` classes.

It has to be noted that, for some observational sets, there is no meaningful definition of an error. Instead, one can define statistical error estimations valid for a certain period, and apply those definitions for observations being performed during said period. That would be the case, for instance, of errors in pointing or focus, where one cannot specify the error for a single observation, but pointing or focus error averages.

AxisFrame.Spatial and Coverage.Spatial

In this subsection, we describe the classes that configure the description of the spatial axis (`AxisFrame.Spatial`), and the characterisation of the coverage in such axis (`Coverage.Spatial`). Figure 7.2 shows the classes and their relationships.

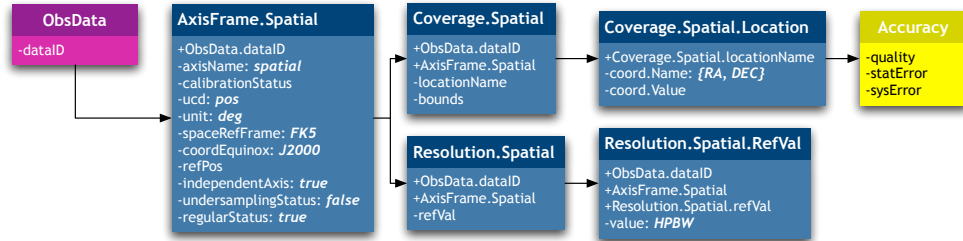


Figure 7.2: Spatial axis frame and coverage metadata; redundant classes for the RADAMS are not shown.

AxisFrame.Spatial: This instance describes the main properties of the Spatial axis, such as calibration status, units (normally, degrees), reference frame, epoch, spatial sampling type and sampling status, etc.

For the DSS63 antenna, the spatial axis is never sampled (because we are not storing maps at this stage), and each scan only needs two pairs of coordinates (plus pointing accuracy) to describe the antenna-pointing pattern.

As the coordinate space is in this case bi-dimensional, and properties for one of the coordinates do not have to be equal for the other, we should either

choose between a vector approach, using two dimensional arrays of coordinates, or splitting the Spatial classes in different subclasses. We choose the first approach, and thus all `AxisFrame.Spatial` attributes will have a space-separated array of values (just two values separated with an space, for a bi-dimensional coordinate space).

If the archive were to be exploited in the third dimension, an additional distance and/or redshift coordinate should be entered. As we have seen, extension in the spatial axis is straightforward. A second option would add an additional axis to the Characterisation data model.

Coverage.Spatial.Location: For the DSS63 antenna, the stored values for this class correspond to the antenna pointing coordinates.

Coverage.Spatial.Bounds: Bounds for spatial data would be the maximum and minimum for spatial coordinates (normally, right ascension and declination) when observing the source. For single point spectroscopic data, this class is redundant, as the antenna does not describe any path across the source.

Coverage.Spatial.Support: Typically, the stored values for this class should be equal to those of `Coverage.Spatial.Bounds`, except when invalid data within `Coverage.Spatial.Bounds` might exist, or in order to describe an elaborate scanning path on the source. This class would describe such a path.

Coverage.Spatial.Sensitivity: In the case of the DSS63 antenna, `Spatial.Sensitivity` would be defined as the beam pattern of the antenna. Another possibility is the combination of beam pattern with receiver efficiency at different elevations.

Bear in mind that `Spatial.Support` for radio astronomical measurements is either a point, for spectroscopic calibrations, or a set of line paths for bolometric and/or continuum mappings. Spatial support will be defined as the set of line paths, and can be described by an array of start/end values for 2D sky coordinates.

For spectroscopic data, this class is redundant, as mentioned above.

AxisFrame.Temporal and Coverage.Temporal

In this subsection, we describe the classes that configure the description of the temporal axis (`AxisFrame.Temporal`), and the characterisation of the coverage in such axis (`Coverage.Temporal`). Figure 7.3 shows the classes and their relationships.

AxisFrame.Temporal: Describes the main properties of the time axis, such as whether the axis is calibrated or not (`calibrationStatus`), units, reference system (`refPos`), time-scale (`timescale`), whether the axis is sampled or not (`undersamplingStatus`), and if sampled, whether sampling is regular or not (`samplingStatus`). In the case of spectroscopic data, time is not sampled.

We will use the `independentAxis` attribute to signal axis dependency as true for continuum observations.

Table 7.1: AxisFrame.Spatial metadata.

Attribute	FITS Keyword	UCD	Description
axisName	assign	meta.id; meta.main	Axis name.
calibrationStatus	assign	obs.calib; meta.code	Calibration status from a controlled vocabulary: uncalibrated , calibrated , relative ^a , normalized ^b .
ucd	assign	meta.ucd; meta.main	Main UCD for the axis.
unit	assign	meta.unit; meta.main	Main units for the axis.
refPos	assign	meta.ref; meta.id	Identification of the origin position within the spaceRef-Frame from a controlled vocabulary; See Space-Time Coordinate Data Model [15].
spaceRefFrame	WCSNAME or RADESYS	pos.frame; meta.id	Identification of the reference system from a controlled vocabulary; see Space-Time Coordinate Data Model [15]: FK4, FK5, ELLIPTIC. . .
coordEquinox	assign	pos; time.equinox	Equinox (only if needed).
epoch	assign	pos; time.epoch	Epoch (only if needed).
independentAxis	assign	pos; obs.param; meta.code	Boolean flag indicating whether the axis is independent of the rest or not.
undersamplingStatus	assign	pos; obs.param; meta.code	Boolean flag indicating whether the data are sampled in this axis or not.
regularStatus	assign	pos; obs.param; meta.code	Boolean flag used in case of sampled data, indicating whether sampling is regular or not.

^a**relative** refers to calibrated data, except for an additive or multiplicative constant.^b**normalized** refers to dimensionless quantities, such as those resulting from the division between two commensurable datasets.

Table 7.2: Coverage.Spatial.Location metadata.

Attribute	FITS Keyword	UCD	Description
coord.name	assign	pos; meta.name	Name of the coordinate whose value goes in coord.value.
coord.value	assign	pos; meta.number	Numeric value for the coordinate coord.name.

Table 7.3: Coverage.Spatial.Bounds metadata.

Attribute	FITS Keyword	UCD	Description
coord.name	assign	pos; meta.name	Name of the coordinate whose value goes in coord.value.
coord.maxValue	assign	pos; meta.number; stat.max	Minimum numeric value for the coordinate coord.name.
coord.minValue	assign	pos; meta.number; stat.min	Maximum numeric value for the coordinate coord.name.

Table 7.4: Coverage.Spatial.Support metadata.

Attribute	FITS Keyword	UCD	Description
code	assign	pos; meta.name	Code for the interval where we will be defining support; an array of [coord.code, startValue, endValue] can be used to define spatial support.
startValue	assign	pos; meta.number	2D array of start values.
endValue	assign	pos; meta.number	2D array of end values.

Table 7.5: Coverage.Spatial.Sensitivity metadata^a.

Attribute	FITS Keyword	UCD	Description
theta[n]	assign	pos.posAng	Theta angle for the nth beam pattern normalised response.
phi[n]	assign	pos.posAng	Phi angle for the nth beam pattern normalised response.
response[n]	assign	arith.factor	Nth normalised beam pattern response.

^aSymmetrical beam patterns could be defined just in one dimension, with pairs of [theta, response] values.

Table 7.6: Coverage.Spatial.Resolution metadata.

Attribute	FITS Keyword	UCD	Description
referenceValue	assign	pos.angResolution	Resolution reference value.

Table 7.7: Accuracy.Spatial metadata.

Attribute	FITS Keyword	UCD	Description
quality	assign	pos; meta.code.qual	Quality code for spatial coordinates; invalid data are flagged with a quality code of 1.
sysError	assign	pos; stat.error.sys	Systematic error for spatial coordinates.
sysErrorHigh	assign	pos; stat.error.sys; stat.max	Maximum systematic error for spatial coordinates.
sysErrorLow	assign	pos; stat.error.sys; stat.min	Minimum systematic error for spatial coordinates.
statError	assign	pos; stat.error	Statistical error for spatial coordinates.
statErrorHigh	assign	pos; stat.error; stat.max	Maximum statistical error for spatial coordinates.
statErrorLow	assign	pos; stat.error; stat.min	Minimum statistical error for spatial coordinates.

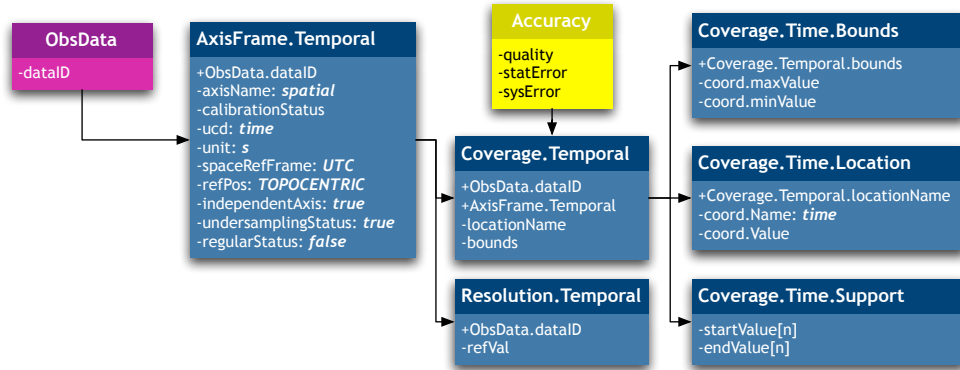


Figure 7.3: Temporal axis and related coverage metadata.

Coverage.Temporal: Collects all the temporal characterisation of the observational data.

Coverage.Temporal.Location: Coverage.Location holds the most representative value for the axis. In the temporal axis, it holds the time of observation, either the starting time or the time at the middle of the observation.

Coverage.Temporal.Bounds: Coverage.Bounds holds the maximum and minimum values for the axis. In the temporal axis, it holds the starting and ending time for the observation.

Coverage.Temporal.Support: Coverage.Support holds the parts of the axis holding valid observational data. In the temporal axis, this is the part of time devoted to flux collection, but just the observation exposition time can be provided.

Coverage.Temporal.Sensitivity: In cases where an instrument had a sensor ramp-up, where some time is needed before 100% sensitivity is achieved, this class would describe such a ramp, and would allow for convenient scaling of data taken during those intervals. If that were never the case, Temporal.Sensitivity would be equal to Temporal.Support.

Resolution.Temporal: Holds the temporal resolution of the data. Most of the time, telescope control system time-stamps are expressed in UTC or LST in seconds, so temporal resolution would be of the order of a second, but normally the instrument could deliver better temporal resolution. We should address this after the initial setup.

We will not develop the **SamplingPrecision.Temporal** class, because we are not sampling the temporal axis.

Accuracy.Temporal: This class has to collect systematic and statistical errors associated with the temporal coordinates of the data. The Quality attribute gives additional information about the temporal data quality. We could use time re-syncing statistics in order to evaluate accuracy.

Table 7.8: AxisFrame.Temporal metadata.

Attribute	FITS Keyword	UCD	Description
axisName	assign	meta.id; meta.main	Axis name.
calibrationStatus	assign	time; obs.calib; meta.code	Calibration status from a controlled vocabulary: uncalibrated , calibrated , relative ^a , normalized ^b
ucd	assign	time; meta.ucd; meta.main	Main UCD for the axis.
unit	assign	time; meta.unit; meta.main	Main units for the axis.
refPos	assign	time; meta.ref; meta.id	Identification of the origin position from a controlled vocabulary.
independentAxis	assign	time; obs.param; meta.code	Boolean flag indicating whether the axis is independent of the rest or not.
undersamplingStatus	assign	time; obs.param; meta.code	Boolean flag indicating whether the data are sampled in this axis or not.
regularStatus	assign	time; obs.param; meta.code	Boolean flag used in case of sampled data, indicating whether sampling is regular or not.
numBins	assign	time; meta.number	Number of time samples.

^a**relative** refers to calibrated data, except for an additive or multiplicative constant.

^b**normalized** refers to dimensionless quantities, such as those resulting from the division between two commensurable datasets.

Table 7.9: Coverage.Temporal.Location metadata.

Attribute	FITS Keyword	UCD	Description
coord.name	assign	time; meta.name	Name of the coordinate whose value goes in coord.value; in this case, time with respect to refPos.
coord.value	assign	time; meta.number	Numeric value for the time location; usually, a MJD or decimal UTC time.

Table 7.10: Coverage.Temporal.Bounds metadata^a.

Attribute	FITS Keyword	UCD	Description
coord.name	assign	time; meta.name	Name of the coordinate whose maximum and minimum values go in coord.maxValue and coord.minValue.
coord.maxValue	assign	time; meta.number; stat.max	Minimum numeric value for the observation time.
coord.minValue	assign	time; meta.number; stat.min	Maximum numeric value for the observation time.

^aThe UCDs for coord.maxValue and coord.minValue could have been, respectively, `time.obs.start` and `time.obs.end`, but we think the proposed UCDs are more consistent with the rest of the axes. Besides, we can defer the election, by using `time.obs.start` and `time.obs.end` at the beginning of the UCD, and appending `meta.number`; `stat.max` or `meta.number`; `stat.min` as additional qualifiers.

Table 7.11: Coverage.Temporal.Support metadata^a.

Attribute	FITS Keyword	UCD	Description
code	assign	time; meta.code	Code for the time interval where we will be defining support; an array of [coord.code, startValue, endValue] can be used to define temporal support.
startValue	assign	time.expo.start	Time interval start value.
endValue	assign	time.expo.end	2DTime interval end value.

^aCoverage.Temporal.Support could be alternatively defined by using just one value, the exposure time, with UCD `time.expo`. The chosen definition, apart from being more consistent across axes, allows for discontinuous temporal support.

Table 7.12: Coverage.Temporal.Resolution metadata.

Attribute	FITS Keyword	UCD	Description
referenceValue	assign	time; meta.number; meta.ref	Resolution reference value.

Table 7.13: Accuracy.Temporal metadata.

Attribute	FITS Keyword	UCD	Description
quality	assign	time; meta.code.qual	Quality code for time coordinates; invalid data are flagged with a quality code of 1.
sysError	assign	time; stat.error.sys	Systematic error for time coordinates.
sysErrorHigh	assign	time; stat.error.sys; stat.max	Maximum systematic error for time coordinates.
sysErrorLow	assign	time; stat.error.sys; stat.min	Minimum systematic error for time coordinates.
statError	assign	time; stat.error	Statistical error for time coordinates.
statErrorHigh	assign	time; stat.error; stat.max	Maximum statistical error for time coordinates.
statErrorLow	assign	time; stat.error; stat.min	Minimum statistical error for time coordinates.

AxisFrame.Spectral and Coverage.Spectral

In this subsection, we describe the classes that configure the description of the spectral axis (`AxisFrame.Spectral`), and the characterisation of the coverage in such axis (`Coverage.Spectral`). Figure 7.4 shows the classes and their relationships.

AxisFrame.Spectral: Describes the main properties of the time axis, such as whether the axis is calibrated or not (`calibrationStatus`), units, reference system (`RefPos`), or the mathematical definition used for the Doppler effect (`DopplerDef`), for velocity calibrated spectra. It is clear that spectral information is indeed sampled, and the number of channels will be coded by `numBins`.

Coverage.Spectral: Collects the different spectral properties of the data.

Coverage.Spectral.Location: `Coverage.Location` holds the most representative value for the axis. In the spectral axis, and being a sampled axis, we will use the frequency for the central sample of the spectrum.

Coverage.Spectral.Bounds: Holds the spectral limits of the data, that is, the starting and ending frequency for the spectrum.

Coverage.Spectral.Support: Holds the spectral support for the data. We could choose to register just the actual bandwidth, or better, to set an array of different intervals where the instrument is sensitive. Better yet, we could register a sensitivity profile for each frequency. We will initially choose the array of intervals as the way to express the spectral support.

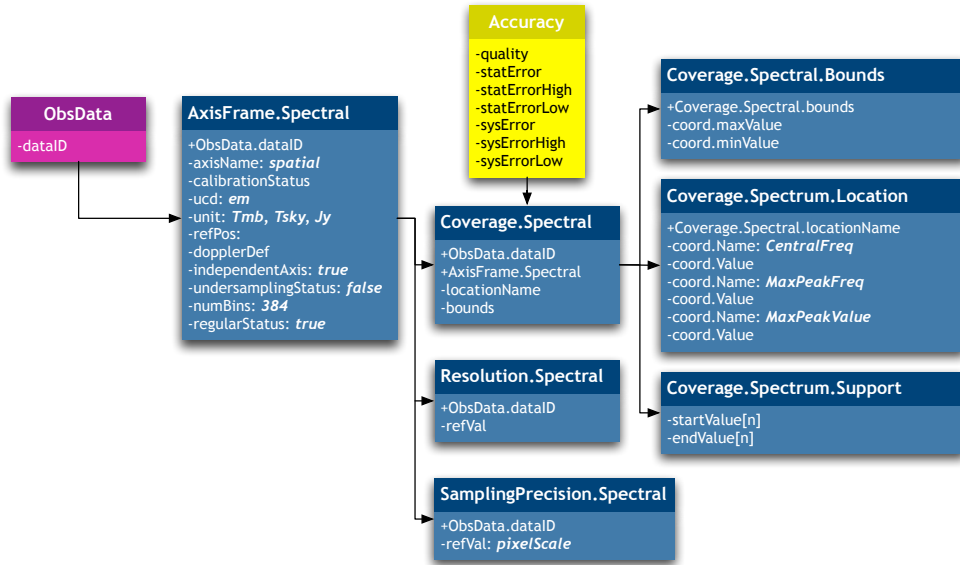


Figure 7.4: Spectral axis and related coverage metadata.

Coverage.Spectral.Sensitivity: Holds the sensitivity profile for the instrument in frequency; in other words, this would be equal to the detailed frequency response of the whole antenna-receiver-backend set.

Resolution.Spectral: Holds the spectral resolution information for the data. For filter banks, it is the filter bandwidth for each filter, or an average filter bandwidth. In the case of spectra obtained from the Fourier transform of the autocorrelation function, it is equal to the bandwidth divided by the number of channels.

SamplingPrecision.Spectral: In the case of the Robledo antenna, this value should be equal to the value stored at **Resolution.Spectral**. For other instruments, **SamplingPrecision.Spectral** defines the frequency steps from one frequency bin to the next.

Accuracy.Spectral: Errors associated with the spectrum. In the same way as the support could change along the spectrum, because of different sensitivities, we might need to characterise the accuracy along the spectrum. At least in the case of Robledo, the SNR is global, and considered equal for all frequencies. If not, a first approximation can be using an average SNR, and maximum and minimum SNR.

AxisFrame.Observable and Coverage.Observable

The Observable axis is, finally, the one directly related to the stored data, instead of data headers. In this subsection, we describe the classes that con-

Table 7.14: AxisFrame.Spectral metadata.

Attribute	FITS Keyword	UCD	Description
axisName	assign	meta.id; meta.main	Axis name (frequency or velocity).
calibrationStatus	assign	em.radio; obs.calib; meta.code	Calibration status from a controlled vocabulary: un-calibrated , calibrated , relative ^a , normalized ^b .
ucd	assign	em.radio; meta.ucd; meta.main	Main UCD for the axis.
unit	TUNITn	em.radio; meta.unit; meta.main	Main units for the axis.
refPos	assign	em.radio; meta.ref; meta.id	Identification of the origin position from a controlled vocabulary.
dopplerDef	VELDEF	spect.dopplerParam; meta.code	Code defining the type of Doppler shift definition used for frequency and/or velocity calibration; from a controlled vocabulary: optical , radio , relativistic .
independentAxis	assign	em.radio; obs.param; meta.code	Boolean flag indicating whether the axis is independent of the rest or not.
undersamplingStatus	assign	em.radio; obs.param; meta.code	Boolean flag indicating whether the data are sampled in this axis or not.
regularStatus	assign	em.radio; obs.param; meta.code	Boolean flag used in case of sampled data, indicating whether sampling is regular or not.
numBins	assign	em.radio; meta.number	Number of spectral samples.

^a**relative** refers to calibrated data, except for an additive or multiplicative constant^b**normalized** refers to dimensionless quantities, such as those resulting from the division between two commensurable datasets.

Table 7.15: Coverage.Spectral.Location metadata.

Attribute	FITS Keyword	UCD	Description
coord.name	assign	em.radio; meta.name	Name of the coordinate whose value goes in coord.value; in this case, frequency with respect to refPos.
coord.value	OBSFREQ	em.radio; em.freq; stat.mean	Numeric value for the central frequency location.

Table 7.16: Coverage.Spectral.Bounds metadata.

Attribute	FITS Keyword	UCD	Description
coord.name	assign	em.radio; meta.name	Name of the coordinate whose maximum and minimum values go in coord.maxValue and coord.minValue.
coord.maxValue	assign	em.freq; meta.number; stat.max	Minimum frequency value.
coord.minValue	assign	em.freq; meta.number; stat.min	Maximum frequency value.

Table 7.17: Coverage.Spectral.Support metadata.

Attribute	FITS Keyword	UCD	Description
code	assign	em.radio; meta.record	Code for the interval where we will be defining support; an array of [coord.code, startValue, endValue] can be used to define spectral support.
startValue	assign	em.freq; stat.min	Frequency interval start value.
endValue	assign	em.freq; stat.max	Frequency interval end value.

Table 7.18: Coverage.Spectral.Sensitivity metadata.

Attribute	FITS Keyword	UCD	Description
numChannels	assign	em.radio; meta.number	Number of channels of the frequency response of the filter that describes spectral sensitivity.
channel[n]	assign	em.freq	Frequency for the nth channel of the filter that describes spectral sensitivity.
response[n]	assign	arith.factor	Filter response for the nth channel.

Table 7.19: Coverage.Spectral.Resolution metadata^a.

Attribute	FITS Keyword	UCD	Description
numChannels	assign	em.radio; meta.number	Number of channels for which resolution is provided.
referenceValue[n]	FREQRES	em.freq; spect.resolution	Resolution reference value for the nth channel.

^aWhen the Sensitivity class is provided, Resolution.numChannels has to be equal to Sensitivity.numChannels, and each channel can support different resolutions. When the Sensitivity class is not provided, Resolution.numChannels has to be set to 1, and Resolution.referenceValue represents the average channel resolution.

Table 7.20: SamplingPrecision.Spectral metadata^a.

Attribute	FITS Keyword	UCD	Description
numChannels	assign	em.radio; meta.number	Number of frequencies that we are sampling.
referenceValue[n]	FREQRES	em.freq; spect.resolution	Resolution reference value for the nth channel.

^aWhen SamplingPrecision.numChannels is set to 0, the number of spectral channels is given by AxisFrame.Spectral.numBins, and SamplingPrecision provides the sampling step in a single referenceValue. Otherwise, SamplingPrecision.Spectral.numBins must be equal to AxisFrame.Spectral.numBins, and when the Sensitivity class is provided, SamplingPrecision.Spectral.numBins must be equal to Spectral.Sensitivity.numChannels, and Spectral.SamplingPrecision.referenceValue[n] equal to Spectral.Sensitivity.channel[n].

Table 7.21: Accuracy.Spectral metadata.

Attribute	FITS Keyword	UCD	Description
quality	assign	em.freq; meta.code.qual	Quality code for frequency coordinates; invalid data are flagged with a quality code of 1.
sysError	assign	em.freq; stat.error.sys	Systematic error for frequency coordinates.
sysErrorHigh	assign	em.freq; stat.error.sys; stat.max	Maximum systematic error for frequency coordinates.
sysErrorLow	assign	em.freq; stat.error.sys; stat.min	Minimum systematic error for frequency coordinates.
statError	assign	em.freq; stat.error	Statistical error for frequency coordinates.
statErrorHigh	assign	em.freq; stat.error; stat.max	Maximum statistical error for frequency coordinates.
statErrorLow	assign	em.freq; stat.error; stat.min	Minimum statistical error for frequency coordinates.

figure the description of the observable axis (`AxisFrame.Observable`), and the characterisation of the coverage in such axis (`Coverage.Observable`). Figure 7.5 shows the classes and their relationships.

AxisFrame.Observable: Describes the main properties of the observable (be it flux, polarization state, or any other), such as whether the axis is calibrated or not (`calibrationStatus`), applicable units, etcetera. Flux information is sampled at the digital conversion stage, and the number of available different flux levels is codified in `numBins`. By specifying units in this class, we define whether the observable data are provided in flux units, or in brightness or antenna temperature. Additional information, such as the polarization, can be accommodated in the same axis in the same way we extended the `Spatial` axis for 2D or 3D coordinates, or by means of (an) additional observable (axis) axes.

Coverage.Observable: Collects all information that directly characterises observable data.

Coverage.Observable.Location: `Coverage.Location` instances hold the most representative value for that axis. In this case, the average spectrum flux is not significant, specially if we have a strong line detection. Hence, for this axis we will take the maximum flux as `Observable.Location`.

Coverage.Observable.Bounds: `Coverage.Bounds` instances record the maximum and minimum values for the axis. In the `Observable` axis, it records minimum and maximum flux for the spectrum.

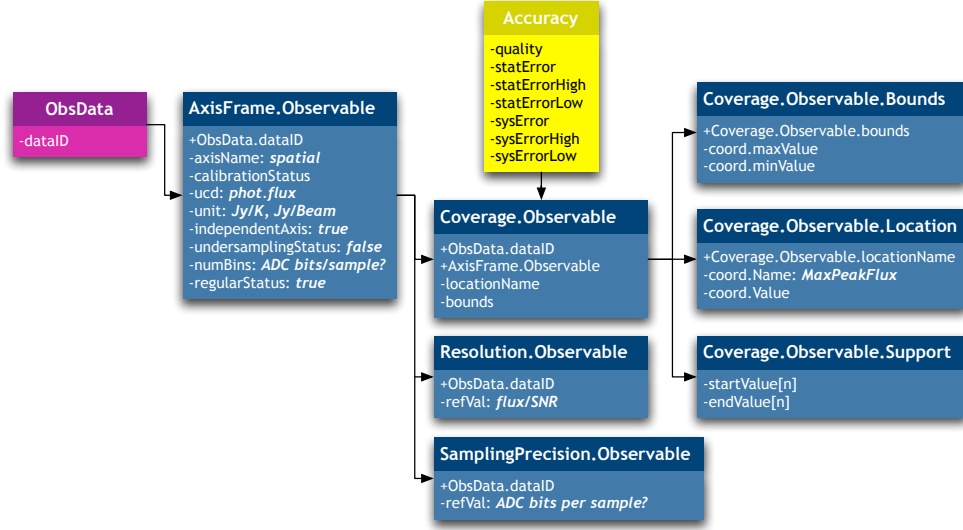


Figure 7.5: Observable axis and related coverage metadata.

Coverage.Observable.Support: This instance should give us the intervals of flux observed in the spectrum, and we could choose between making it equal to **Coverage.Observable.Bounds**, or defining an interval that holds a given dispersion from the **Coverage.Observable.Location** value. We will initially consider **Coverage.Observable.Support** as equal to **Coverage.Observable.Bounds**.

Coverage.Observable.Sensitivity: This would store the change rate in the observable axis for a given observed flux.

Resolution.Observable: We keep the flux resolution for the data, defined as mean flux divided by the SNR, or mean noise.

SamplingPrecision.Observable: Instances of this class indicate the sampling precision. For the observable (flux) axis, this should be equal to the Analog to Digital Converter (ADC) resolution, or worse if more processing is involved, and should be tabulated by instrument setup. We might also consider this axis as non-sampled, as **Resolution.Observable** is usually well above ADC resolution, and drop this class.

Accuracy.Observable: Collected flux associated errors. Errors on the observable axis can be dependent on received flux, and so we should give an average value, and either variance or maximum and minimum accuracy on the observable axis.

If the observable variable were a different one, the main difference would correspond to the change of UCDs. In the particular case of polarization, **phot.flux** would be replaced by **phys.polarization**. Other changes would include an additional characterization of the parameters been actually returned by the instrument, and their relationship with Stokes parameters.

Table 7.22: AxisFrame.Observable metadata (when the observed variable is flux).

Attribute	FITS Keyword	UCD	Description
axisName	assign	meta.id; meta.main	Axis name.
calibrationStatus	assign	em.radio; obs.calib; meta.code	Calibration status from a controlled vocabulary: uncalibrated , calibrated , relative^a , normalized^b .
ucd	assign	phot.flux; meta.ucd; meta.main	Main UCD for the axis.
unit	TUNITn	phot.flux; meta.unit; meta.main	Main units for the axis.
refPos	assign	phot.flux; meta.ref; meta.id	Identification of the origin position from a controlled vocabulary.
independentAxis	assign	phot.flux; obs.param; meta.code	Boolean flag indicating whether the axis is independent of the rest or not.
undersamplingStatus	assign	phot.flux; obs.param; meta.code	Boolean flag indicating whether the data are sampled in this axis or not.
regularStatus	assign	phot.flux; obs.param; meta.code	Boolean flag used in case of sampled data, indicating whether sampling is regular or not.
numBins	assign	phot.flux; meta.number	Number of spectral samples (if the axis is sampled)

^a**relative** refers to calibrated data, except for an additive or multiplicative constant

^b**normalized** refers to dimensionless quantities, such as those resulting from the division between two commensurable datasets.

Table 7.23: Coverage.Observable.Location metadata (when the observed variable is flux).

Attribute	FITS Keyword	UCD	Description
coord.name	assign	phot.flux; meta.name	Name of the coordinate whose value goes in coord.value; in this case, flux with respect to refPos.
coord.value	assign	phot.flux; stat.max	Numeric value for the maximum flux (flux location).

Table 7.24: Coverage.Observable.Bounds metadata.

Attribute	FITS Keyword	UCD	Description
coord.name	assign	phot.flux; meta.name	Name of the coordinate whose maximum and minimum values go in coord.maxValue and coord.minValue.
coord.maxValue	assign	phot.flux; stat.max	Minimum flux value.
coord.minValue	assign	phot.flux; stat.min	Maximum flux value.

Table 7.25: Coverage.Observable.Support metadata (when the observed variable is flux).

Attribute	FITS Keyword	UCD	Description
code	assign	em.radio; meta.record	Code for the interval where we will be defining support; an array of [coord.code, startValue, endValue] can be used to define spectral support.
startValue	assign	em.freq; stat.min	Frequency interval start value.
endValue	assign	em.freq; stat.max	Frequency interval end value.

Table 7.26: Coverage.Observable.Resolution metadata.

Attribute	FITS Keyword	UCD	Description
referenceValue	assign	phot.flux; stat.snr; arith.ratio	Flux resolution (Flux/SNR), as $\frac{\text{Flux}}{\text{SNR}} = \text{flux} \times \frac{\text{fluxNoise}}{\text{fluxSignal}}$. Equivalent to average noise.

Table 7.27: SamplingPrecision.Observable metadata^a.

Attribute	FITS Keyword	UCD	Description
referenceValue	assign	phot.flux; instr.precision	Flux sampling precision; maybe expressed as the number of bits per sample.

^aSomewhat redundant, given the presence of AxisFrame.Observable.numBins. However, we include it for completeness.

Table 7.28: Accuracy.Observable metadata.

Attribute	FITS Keyword	UCD	Description
quality	assign	phot.flux; meta.code.qual	Quality code for the observed flux; invalid data are flagged with a quality code of 1.
sysError	assign	phot.flux; stat.error.sys	Systematic error for the observed flux.
sysErrorHigh	assign	phot.flux; stat.error.sys; stat.max	Maximum systematic error for the observed flux.
sysErrorLow	assign	phot.flux; stat.error.sys; stat.min	Minimum systematic error for the observed flux.
statError	assign	phot.flux; stat.error	Statistical error for the observed flux.
statErrorHigh	assign	phot.flux; stat.error; stat.max	Maximum statistical error for the observed flux.
statErrorLow	assign	phot.flux; stat.error; stat.min	Minimum statistical error for the observed flux.

7.3. Provenance

This data model provides support for the description of the data generation model for a particular observation. This includes not only the instrumental configuration, but also a specification of the type of processes/pipelines the data have been through, and of the weather conditions. We will subdivide this class in several sub-data-models, for easier understanding of the Provenance data model.

Provenance.Instrument

Figure 7.6 shows the classes associated with the instrumental configuration for the observation.

InstrumentConf: Each observation is associated to a particular instrumental configuration, which results from the particular configuration of the instrument + antenna + feed system. InstrumentConf instances group those settings.

Instrument: Instances of this class specify the instrument configuration, as used for the observation.

Instrument.Location: This is an instance of a Location class, used for specifying the location for the instrument.

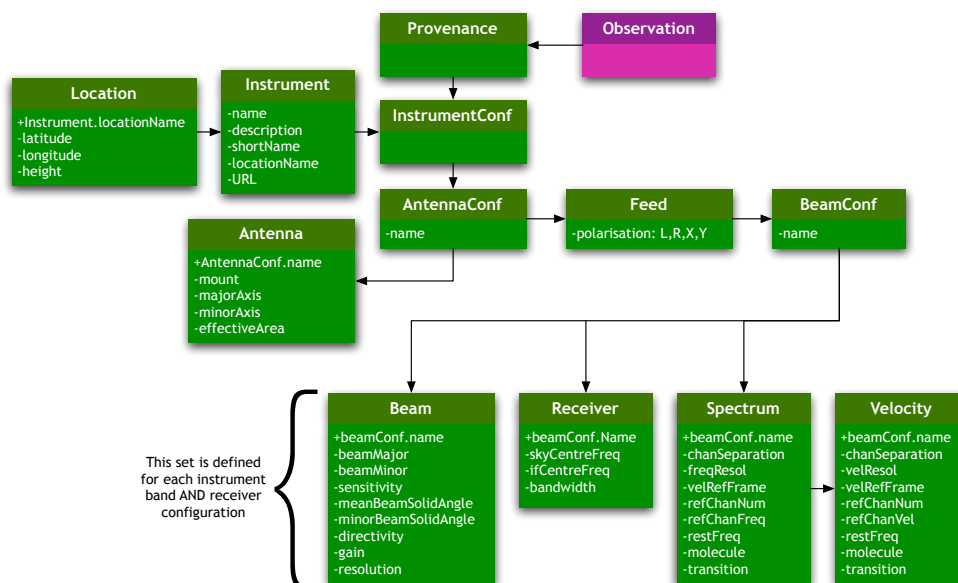


Figure 7.6: Provenance.Instrument data model.

AntennaConf: In the same way InstrumentConf allows the grouping for all the instrumental settings, AntennaConf instances group together Antenna and Feed info (regarding polarization), plus BeamConf —another aggregator or class—. Several AntennaConf instances can provide information for different antennas and feeds. Each possible antenna configuration will be labelled by a name.

Antenna: Instances of this class specify the general properties for each given antenna. We will also use these instances to specify the type of scan being performed on the source from a controlled vocabulary.

Feed: Instances of this class —one or more per AntennaConf— specify each of the feed horns used for the observation, and their corresponding polarization from a controlled vocabulary: L, R, X, Y.

BeamConf: This class is used to group antennas, feeds and beams. The relationship between BeamConf and Feed instances allows the specification of different beams, formed by the combination of different feeds. In a single-dish single-feed configuration, there is only one beam associated to a given receiver. We can also use this association for a single-dish, multiple-feed configuration where each feed goes to a different receiver.

Beam: Metadata for this class specifies the actual beam for the telescope, associated to a given spectral band.

Receiver: This metadata are used to describe the most relevant properties of the receiver, such as receiver type, intermediate frequency —in the case of

Table 7.29: Provenance instrument metadata.

Attribute	FITS Keyword	UCD	Description
name	INSTRUME	meta.id; instr; meta.main	Instrument name.
description	assign	meta.note; meta.main	Instrument description.
shortName	assign	instr; meta.id	Short name for the instrument.
locationName	assign	instr; pos; meta.id	Localization of the instrument.
URL	COMMENT	instr; meta.ref.url	URL for the instrument (website for the instrument, documentation, or any other type of instrument description).

Table 7.30: Instrument location metadata.

Attribute	FITS Keyword	UCD	Description
locationName	assign	instr; pos; meta.id	Name of a particular instrument location.
latitude	SITELAT	instr; pos.earth.lat	Instrument location latitude.
longitude	SITELONG	instr; pos.earth.lon	Instrument location longitude.
altitude	SITELEV	instr; pos.earth.altitude	Instrument location altitude.

heterodyne stages—, etc.

Spectrum: In case of spectroscopic observations, the spectral analyser that has been used is specified by instances of this class.

Velocities: This class mirrors the Spectrum class, and is preferred for those cases where we used velocities, instead of frequency.

Provenance.AmbientConditions

The Provenance.AmbientConditions subclass deals with all the metadata needed to specify weather conditions, air mass, opacity, etc. Figure 7.7 shows the corresponding classes and relationships.

AmbientConditions: Holds all metadata related with weather conditions for the observation, such as humidity, wind speed, opacity at zenith, etc.

OpacityCurve: Includes the opacity curve (linked as a VOTable file) associated to the observing term where the data were observed (which we will

Table 7.31: Antenna configuration metadata.

Attribute	FITS Keyword	UCD	Description
name	assign	instr.telescope; meta.title; meta.id	Name of the particular antenna.
scanType	assign	instr.setup; meta.code	Type of scan being performed by this antenna, from a limited vocabulary (suggested by the RDM [8]): beamSwitch, cal, cross, dopplerTrack, dwell, focus, frequencySwitch, holography, mosaic, onOff, onTheFly, point, positionSwitch, pulsar, raster, skyDip, tiedArray, track, wobblerSwitch.
mount	assign	meta.note	Mount type for the telescope from a limited vocabulary: azimuthal, equatorial, altazimuthal, dobson, german equatorial.
majorAxis	assign	instr; phys.size.smajAxis	Major axis dimensions.
minorAxis	assign	inst; phys.size.sminAxis	Minor axis dimensions.
effectiveArea	assign	instr; phys.area	Effective instrument area.

Table 7.32: Feed configuration metadata.

Attribute	FITS Keyword	UCD	Description
polarization	STOKES	pos.posAng; phys.polarization; meta.code	Polarization value from a controlled vocabulary: L, R, X, Y

Table 7.33: Beam configuration metadata.

Attribute	FITS Keyword	UCD	Description
beamMajor	BMAJ/HPBW	<code>instr.beam;</code> <code>phys.size.smajAxis</code>	Major axis HPBW of the main lobe of the beam.
beamMinor	BMIN/HPBW	<code>instr.beam;</code> <code>phys.size.sminAxis</code>	Minor axis HPBW of the main lobe of the beam.
sensitivity	BEAMEFF	<code>instr.beam;</code> <code>instr.sensitivity</code>	Beam average sensitivity.
mainBeamSolidAngle	assign	<code>instr.beam;</code> <code>pos.posAng;</code> <code>meta.main</code>	Main lobe's beam solid angle.
totalBeamSolidAngle	assign	<code>instr.beam;</code> <code>pos.posAng; stat.max</code>	Total beam solid angle, including secondary lobes.
directivity	assign	<code>instr.beam;</code> <code>instr.setup;</code> <code>arith.factor</code>	Directivity percentage.
gain	ANTGAIN	<code>instr.beam;</code> <code>instr.setup;</code> <code>arith.factor</code>	Beam gain ^a .

^aWe still have to clarify if the gain attribute is related to the directivity concept or not, and if it is related with the receiving stages or not.

Table 7.34: Receiver metadata.

Attribute	FITS Keyword	UCD	Description
type	BACKEND	<code>instr.setup;</code> <code>meta.note</code>	Receiver type (HEMT, Bolometer...).
skyCentreFreq	assign	<code>src; em.radio;</code> <code>em.freq</code>	Antenna tuning frequency.
IFCentreFreq	assign	<code>instr.setup;</code> <code>em.freq</code>	Heterodyne receiver intermediate frequency (or list of frequencies).
bandwidth	BANDWID	<code>instr.bandwidth</code>	Filter-bank total bandwidth.

Table 7.35: Spectrum metadata. It might be necessary to change the MOLECULE and TRANSITI keywords by LINE, for better CLASS compatibility.

Attribute	FITS Keyword	UCD	Description
numChannels	NAXISn	spect; em.freq; meta.number	Number of spectral channels.
chanSeparation ^a	assign	spect; em.freq	Mean channel separation (in frequency units), or channel frequency separation array.
freqResolution	FREQRES	spect.resolution; em.freq	Frequency resolution.
refChanNum	assign	spect; em.freq; meta.number; meta.ref	Spectral reference channel.
refChanFreq	OBSFREQ	spect; em.freq; meta.code; meta.ref	Spectral reference frequency (observed frequency).
restFreq	FREQn or RESTFREQ	spect.line; em.freq	Observed spectral line rest frequency.
molecule	MOLECULE ^b	spect; phys.mol; meta.id	Molecule name.
transition	TRANSITI ^c	spect; phys.atmol.transition; meta.id	Transition.

^aThere is a certain redundancy between the Provenance.Spectrum.chanSeparation attribute and the Coverage.Spectral.Resolution attributes.

^bIt might be necessary to change the MOLECULE keyword by LINE, for better CLASS compatibility.

^cIt might be necessary to change the TRANSITI keyword by LINE, for better CLASS compatibility.

derive from the nearest two skydip scans performed before and after the observation). We propose the inclusion of an array of [elevation, Tsky] pairs, together with the azimuth and the starting time of the skydip. Calculation of the opacity curve is different for bolometric or heterodyne observations. It is also necessary to include information on the atmospheric model and/or software used for opacity fitting (ATM is the model used by MOPSIC, at the IRAM 30m antenna).

Provenance.Processing

This class allows us to specify the type of processing applied to the data before storage, including some processes that conform the observation itself, such as determination of frequencySwitching or positionSwitching for source data comparison. We use just two classes, Processing and Calibration —this is a subclass of processing—. Order is relevant, and it should be possible

Table 7.36: Velocity metadata.

Attribute	FITS Keyword	UCD	Description
numChannels	assign	spect; phys.veloc; meta.number	Number of velocity channels.
chanSeparation ^a	assign	spect; phys.veloc	Mean channel separation (in velocity units), or velocity channel separation array.
velResolution	assign	spect.resolution; phys.veloc	Velocity resolution.
velRefFrame	assign	spect; phys.veloc; pos.frame; meta.id	Identification of the reference system used for the velocity.
refChanNum	assign	spect; phys.veloc; meta.number; meta.ref	Velocity reference channel.
refChanFreq	assign	spect; phys.veloc; meta.code; meta.ref	Frequency for the velocity reference channel.
restFreq	RESTFREQ	spect.line; em.freq	Observed spectral line rest frequency.
molecule	MOLECULE ^b	spect; phys.mol; meta.id	Molecule name.
transition	TRANSITI ^c	spect; phys.atmol.transition; meta.id	Transition.

^aThere is a certain redundancy between the Provenance.Velocity.chanSeparation attribute and the Coverage.Spectral.Resolution attributes.

^bIt might be necessary to change the MOLECULE keyword by LINE, for better CLASS compatibility.

^cIt might be necessary to change the TRANSITI keyword by LINE, for better CLASS compatibility.

to reconstruct the pipeline by the ordering of Processing and/or Calibration instances. Figure 7.8 shows these classes.

Processing: It holds information specifying the type of processing applied to data before archival. This includes pseudo-observational techniques such as position switching or frequency switching, as well as the type of data averaging, data weighting, etc. Table 7.39 provides minimal initial metadata, using arrays of parameter keywords for extensibility at the expense of complexity.

Calibration: It is a subclass of Processing, where the type of processing is dataCalibration. In this class there are additional attributes to specify the type of calibration, and the axes to where this calibration will be applied. Table 7.40 provides minimal initial metadata, using arrays of parameter keywords for extensibility at the expense of complexity.

We still have to develop a calibration and/or pointing model; maybe based upon IRAM-Multi-Beam-FITS, or GBT FITS calibration tables.

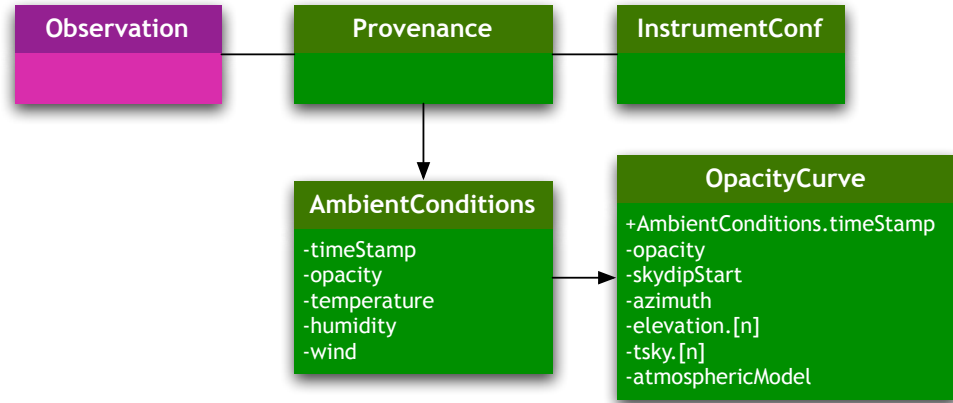


Figure 7.7: Provenance.AmbientConditions data model.

Table 7.37: AmbientConditions metadata.

Attribute	FITS Keyword	UCD	Description
opacity	TAUZEN	phys.absorption.coeff	Opacity at zenith estimated at the observation frequency.
airMass	AIRMASS	obs.airMass	Air mass at zenith at the observing site.
temperature	TAMBIENT	phys.temperature	Ambient temperature.
humidity	HUMIDITY	obs.atmos; phys.columnDensity	Ambient humidity.
waterVapour	TAU_WPATH_RD<freq>	obs.atmos; phys.pressure	Equivalent pressure of the water vapour column.
tauFrequency	TAU_WPATH_RD<freq>	obs.atmos; em.freq	Tau radiometer frequency.
wind	WINDSPEE	obs.atmos; phys.veloc	Wind speed.

7.4. Target

It is necessary to have a systematic storage of the targets being studied, because we have to allow queries by previously studied targets, as it is very likely that query will be the most used one.

The DMO [5] differentiates between several types of targets:

- **Astronomical object:** It is a target for which some properties are known before the observation.
- **Source:** An entity created after analyzing an observation; represents

Table 7.38: Opacity metadata.

Attribute	FITS Keyword	UCD	Description
opacity	TAUZEN	phys.absorption.coeff	Opacity at zenith estimated at the observation frequency.
skydipStart	DATE-OBS	time.obs.start	Skydip starting time.
azimuth	AZIMUTH	pos.az	Skydip azimuth.
elevation[n]	ELEVATIO	pos.el	Skydip scan elevation.
tsky[n]	assign	instr.skyTemp	Sky temp at n th skydip.
atmosModel	assign	meta.modelled; obs.atmos; meta.id	Atmospheric model identification.

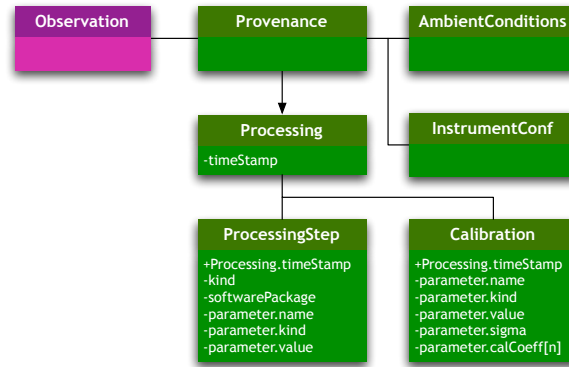


Figure 7.8: Provenance.Processing data model.

the actual detection by an instrument.

- **Field:** A region of the sky, not a single point or collection of points.
- **Pointing target:** A particular location for an observation. It can be part of an Astronomical object, a Field, or other entities, such as a Calibrator.

IVOA has not developed the Target class as a whole, but the authors of the SDM [7] have developed a set of possible metadata. We will initially adopt that set for the RADAMS, and we can see the associated metadata in Figure 7.9.

Another data model dealing with Targets is the Resource Data Model [16], which considers additional targets for observations, such as radio flux calibrators, pointing calibrators, etc.

We have to take into account that most of the time the archive will store ObsData associated to Pointing targets, but said Pointing targets can also be

Table 7.39: Processing Step metadata.

Attribute	FITS Keyword	UCD	Description
timestamp	DATE-RED	obs.param; time.epoch	Timestamp for the processing step being performed.
type	assign	obs.param; meta.code	Type of processing applied to source data; comes from a controlled vocabulary: <code>unprocessed</code> , <code>noiseWeightedAverage</code> , <code>nonWeightedAverage</code> .
softwarePackage	assign	meta.software; meta.id	Software package used for data processing; should come from a controlled vocabulary: <code>CLASS</code> , <code>AIPS</code> , <code>AIPS++</code> , <code>CASA</code> , <code>MOPSIC</code> , <code>GILDAS</code> , <code>MIRA</code> , <code>MIR</code> , <code>other</code> . In the case of <code>other</code> , the actual package that was used should be added as a parameter, with parameter.name as <code>softwarePackage</code> and the parameter.value as the package name.
parameter[n].name	assign	obs.param; meta.code	Additional processing parameter name, whose value will be in parameter.value; eventually, we will have a controlled list of possible parameter.name values.
parameter[n].type	assign	obs.param; meta.code	From a controlled vocabulary: <code>integer</code> , <code>float</code> , <code>string</code> ... At least all of FITS data types should be present.
parameter[n].value	assign	obs.param ^a	Value for the parameter indicated by parameter.name.

^aThe final UCD to mark parameter[n].value will be calculated when writing the VOTable, as it depends on parameter.type; it will be `obs.param; meta.number` most of the time, but it could be `obs.param; meta.name` or `obs.param; meta.code`, depending on the context.

Table 7.40: Calibration metadata^a.

Attribute	FITS Keyword	UCD	Description
timestamp	DATE-RED	obs.param; time.epoch	Timestamp for the calibration step being performed.
parameter.name	assign	obs.calib; obs.param; meta.id	Keyword defining the parameter that we will characterise with the remaining attributes.
parameter.type	assign	obs.calib; obs.param; meta.code	Type of calibration parameter used, from a controlled vocabulary: additive , factor , polynomial , exponential , logarithmic .
parameter.value	assign	obs.calib; obs.param; meta.number	Value for the main calibration parameter, where parameter.type is not polynomial .
parameter.sigma	assign	obs.calib; obs.param; meta.number	Value of sigma, for exponential calibrations.
parameter.calCoeff.[n]	assign	obs.calib; obs.param; meta.number	n th degree coefficient for a polynomial calibration parameter; polynomial degree is derived from the maximum n.

^aIt is mandatory that at least one [parameter.name, parameter.type, parameter.value] triplet appears, with `fluxScale` as parameter.name, and one of `antennaTemperature`, `mbBrightnessTemperature`, or `S_nu` as the parameter.value, with a parameter.type of `string`.

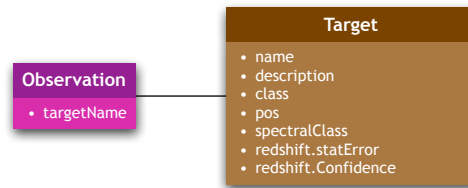


Figure 7.9: Target data model.

related to Astronomical Objects. E.g., in the scientific case we mentioned in Section 7.1, the dish is pointed to a Pointing target in order to find a maser, but that Pointing target will be most likely included within an Astronomical object, such as a planetary nebula. The association of the Astronomical object and the Pointing target will be made by means of the Target class.

7.5. Packaging

The Packaging class is used to specify how data from different sources will be presented together. For instance, if we wanted to retrieve data belonging to our already mentioned maser survey, a VO system could reply with a Multi-Beam FITS file containing all the scans and sub-scans that conform the On-Off patterns for a single issued observation, or with a `.zip` file with all the FITS files belonging to the survey, or with just a single On-Off pair, etc.

An instance of the Packaging class would describe the contents of the data retrieved, in terms of project organisation, and of the particular files being actually delivered.

Unfortunately, there is no Packaging class defined at the VO level, so we have to resort to other packaging description mechanisms available in other archiving tools.

First, we have to state the requirements for this class:

- The Packaging class will describe the type of compression or grouping, if any, applied to the data. So, possible results should form a controlled vocabulary, such as `FITS`, `VOTable`, `zip`, `tar`, `gz`, `bz`, `bz2`, `tgz`, `tbz`, `tbz2`, corresponding to: raw files —a FITS file or a VOTable—, a zip file, a tar file, a raw file compressed by gzip, bzip, bzip2, and a gzipped, bzippped or bzippped2 tar file.
- We believe a standard VO packaging scheme is needed in order to facilitate distribution. We propose one such scheme, that we call VOPack —Virtual Observatory Package; `.vopack` file extension—. We will define the VOPack as a “tarred and gzipped” folder (`.tgz` file) with an XML content descriptor, which acts as VO-aware manifest of contained assets. The VOPack is further described in Appendix D.

We will use the MPEG-21 Digital Item Description as an inspiration for the VOPack.

7.6. Policy

The role of the Policy data model is to allow for very different policies to be applied to the data. In the case of a VO archive where the data are not immediately available (they have to be manually incorporated to the archive), Policy becomes simpler, as in “everything in the archive is available for everyone”. Figure 7.10 shows the different classes needed to characterise the archive policy.

Policy instances attached to raw data shall need to identify the different roles of agents —not just people, but software packages too— accessing the archive, and give them appropriate access rights to pieces of data.

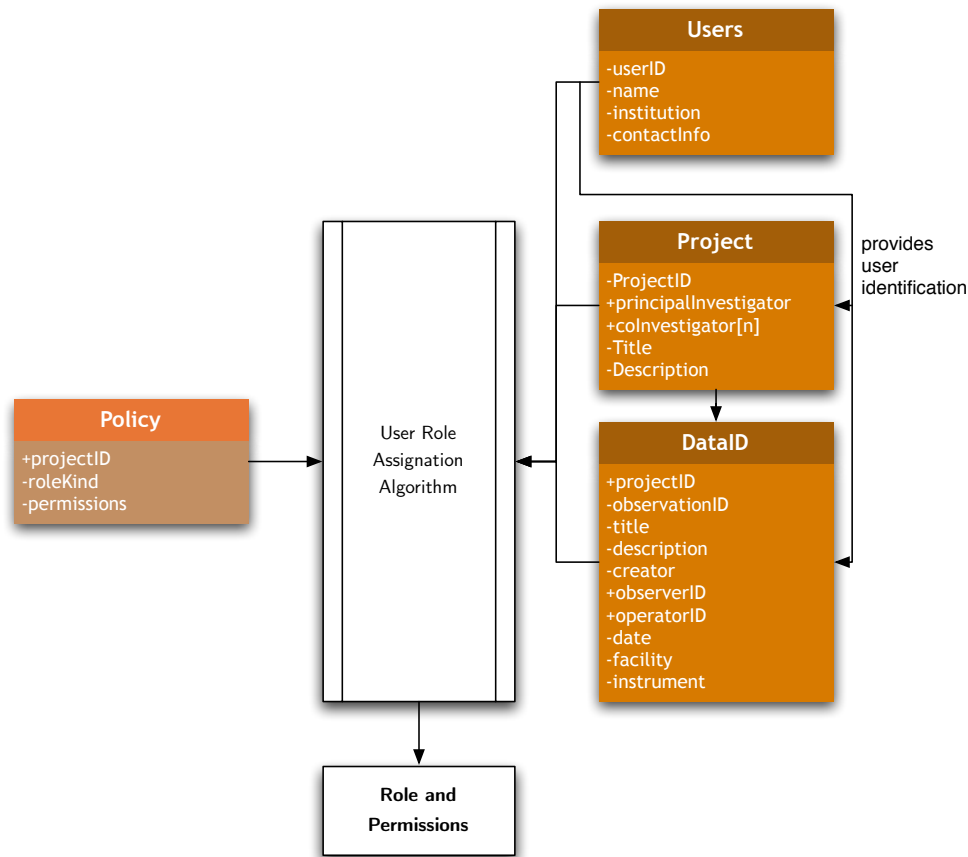


Figure 7.10: Policy data model, with role and permissions.

In the case of the Robledo Archive, the policy to be implemented should be the standard NRAO policy: 18 months since the end of the observations. Such a simple policy is better accomplished by not entering data into the archive until the proprietary period has expired.

A more flexible solution would make use of an array of permissions by role, where roles are derived from the relationship between the principal investigator and/or the observer, and observatory staff.

Such roles would be chosen from a controlled vocabulary (`principalInvestigator`, `observer`, `coInvestigator`, `observatoryStaff`, `none`); those roles are derived from user/project relationships, so that people not belonging to the observatory, and which have nothing to do with the project, would default to `none`.

Each project in the archive should have, at least, an explicit policy of what is allowed for some with none relationship with the Principal Investigator, and people with `principalInvestigator` roles have all access rights to the

Table 7.41: Policy metadata.

Attribute	FITS Keyword	UCD	Description
projectID	PROJID	<code>meta.curation.project;</code> <code>meta.id</code> ^a	Project identifier.
roleKind	assign	<code>meta.policy.role;</code> <code>meta.code; meta.id</code> ^b	Role kind for which permissions will be provided.
permissions	assign	<code>meta.policy.permissions;</code> <code>meta.code</code> ^a	Permissions provided for roleKind users of this project. The particular permissions to be provided are to be discussed by IVOA.

^aThere is no `meta.curation.project` UCD, but we propose the inclusion of one.

^bThere are no `meta.policy.*` UCDs, but we propose, at least, the inclusion of the `meta.policy` atom.

Table 7.42: Policy related Users metadata.

Attribute	FITS Keyword	UCD	Description
userID	COMMENT	<code>meta.id</code>	User identifier for all user related operations in the archive.
name	COMMENT	<code>meta.name</code>	Real name of the user. Any known user of the archive has to be registered, or be anonymous.
institution	COMMENT	<code>meta.name</code>	Name of the institution to which the user belongs.
contactInfo	COMMENT	<code>meta.note</code>	Contact info (probably e-mail) for this user.

archive; people with roles other than `principalInvestigator` would fallback to the `none` role, if their role's permissions are not explicitly declared.

We will provide Policy metadata in Table 7.41, but also we will specify a subset of Curation attributes needed for successful Policy attribution.

We are also considering a data-oriented policy, instead of a user-oriented policy. Data-oriented policies allow each datum —possibly by means of the `DataID` or `Project` classes— to be searched and found, but no additional information, or only partial information —possibly having to do with the owner of the datum— can be retrieved, depending on the data policy in place. This needs at least an additional attribute in the `DataID` class.

Table 7.43: Policy related Project metadata.

Attribute	FITS Keyword	UCD	Description
projectID	PROJID	<code>meta.curation.project</code> ; <code>meta.id</code> ^a	Project identifier.
principalInvestigator	COMMENT	<code>meta.id</code>	User identifier for the principalInvestigator of the project.
coInvestigator[n]	COMMENT	<code>meta.id</code>	User identifier for each of n project coInvestigators.
title	COMMENT	<code>meta.curation.project</code> ; <code>meta.title</code> ^a	Project title.
description	COMMENT	<code>meta.curation.project</code> ; <code>meta.note</code> ^a	Project description.

^aThere is no `meta.curation.project` UCD, but we propose the inclusion of one.

7.7. Curation

The Curation data model groups all metadata mandatory for resources to be published by a VO Registry. Several additional classes are needed for particular instances of data. Such classes and metadata are described by “Resource Metadata for the VO”, an IVOA Recommendation [16]. Figure 7.11 encompasses all classes related to metadata for curation.

Observation: The **Observation** class is was discussed earlier. All observations are related to a single **Curation** class.

Curation: This is the main class that will group all curation related metadata with relationships to complementary classes such as **DataID** and the rest. It is the same for all data contained in the VO resource (archive) being described.

DataID: There is an instance of this class for each different piece of data stored, or at least for each different way to store data. The Spectral Data Model advocates for this separation.

ObservingProgram: Instances of this class keep common information about different observations, describing its scientific or technical goals. They also specify which is the project and/or proposal to which this observing program belongs. This class allows for easy querying by project or by common goals, such as maser surveys.

While there is debate about whether this class belongs to the Curation or to the Provenance Data Model, we believe it belongs to the Curation Data Model, as it is an organisational unit, and in our view it has nothing to do with how the data were taken.

Project: This class is the main class from an organizational point of view. All observations belong to a single project through a single proposal. A **Project**

Table 7.44: Policy related DataID metadata.

Attribute	FITS Keyword	UCD	Description
projectID	PROJID	<code>meta.curation.project</code> ; <code>meta.id</code> ^a	Project identifier.
observationID	OBSID	<code>obs</code> ; <code>meta.dataset</code> ; <code>meta.id</code>	User identifier for the principalInvestigator of the project.
coInvestigator[n]	COMMENT	<code>obs</code> ; <code>meta.id</code>	User identifier for the n^{th} project coInvestigator.
title	COMMENT	<code>meta.curation.project</code> ; <code>meta.title</code> ^a	Project title.
description	COMMENT	<code>meta.curation.project</code> ; <code>meta.note</code> ^a	Project description.
creator	AUTHOR	<code>meta.curation</code> ; <code>meta.id</code>	User identifier for the creator of the data entry.
observerID	OBSERVER	<code>obs.observer</code> ; <code>meta.id</code>	User identifier for the person performing the observation producing this data entry.
operatorID	OBSERVER	<code>obs.operator</code> ; <code>meta.id</code> ^b	User identifier for the person performing operator duties while performing the observation.
date	DATE-OBS	<code>time.obs.start</code>	Date of observation.
facility	TELESCOP	<code>instr.obsty</code>	Facility (observatory) where the telescope/instrument resides in.
instrument	INSTRUME	<code>instr.tel</code>	Instrument performing the observation ^c .

^aThere is no `meta.curation.project` UCD, but we propose the inclusion of one.^bWe propose the inclusion of either `obs.operator` or `instr.operator` as new UCDs to characterise operator-related data. However, `obs.observer` can be used when providing both observer and operator at the same time.^cWe have to study whether this should contain an instrument-backend pair, or have different attributes for both.

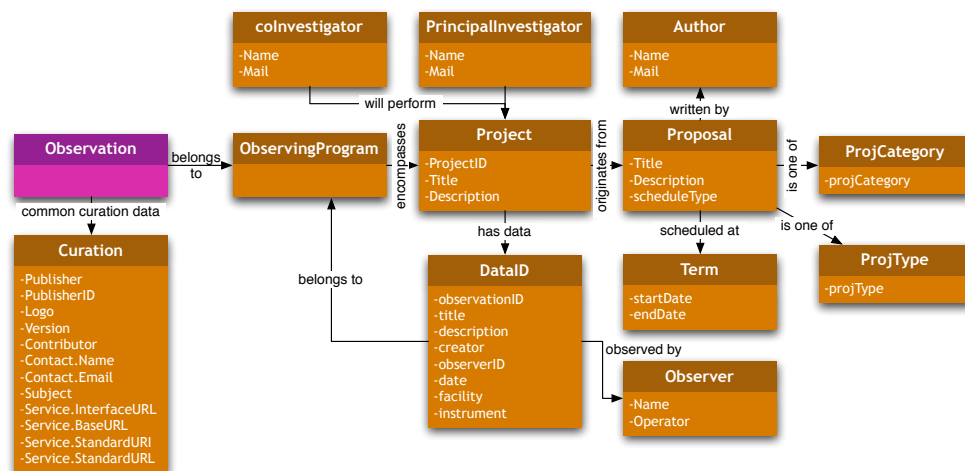


Figure 7.11: Curation data model.

instance is related to a collection of **Proposal** instances.

Proposal: As stated in the **Project** class, a **Project** can be divided in a series of **Proposals**. A **Proposal** instance can be related to one or more observations. No observation can be related to more than one **Proposal** instance.

Author: An **Author** instance contains information about **Proposal** authors.

Observer: **Observer** instances represent the observer/s assigned to a **Proposal**. It also contains a relationship with a **Contact** acting as the operator for the instrument.

ProjCategory: This instance specifies the scientific category of the project, from a controlled vocabulary; possible values, as specified by the Resource Data Model, are: *instrumental*, *galactic*, *terrestrial*, *solarSystem*, *extragalactic*, *stellar*...

ProjType: This instance specifies the type of observation performed for a particular proposal. Possible values are specified by the Resource Data Model, and form a controlled vocabulary derived from ATNF and NRAO proposals: *astrometry*, *bandwidthSynthesis*, *circularPolarization*, *continuum*, *engineering*, *fillerTime*, *highTimeResolution*, *imaging*, *instrumental*, *linearPolarization*, *mapping*, *monitoring*, *mosaic*, *multibeam*, *point-Source*, *phasedArray*, *polarimetry*, *snapshot*, *solar*, *spectroscopy*, *survey*, *timeBinning*.

Term: Specifies start and end dates (ISO dates, with time stamp) for the proposal *observing term* or *scheduling block*.

Capítulo 8

Conclusiones y trabajo futuro

En esta memoria se ha mostrado que:

- Se pueden utilizar múltiples tecnologías multimedia en archivos astronómicos, tanto para proceso de datos, como para minería y archivo.
- Si bien los estándares de modelos de datos y archivo de IVOA son un buen punto de partida, ha sido necesario escoger entre diferentes estándares y propuestas.
- Además, es necesario proporcionar un nivel de detalle superior en muchos de los modelos de datos propuestos para que sean útiles como base para la implementación de un archivo.

En particular, en esta memoria se ha llevado a cabo:

- Un cuidadoso estudio de todos los estándares de IVOA.
- La selección de los estándares necesarios para el desarrollo de un archivo radioastronómico para datos espectroscópicos en antenas de plato único, y el establecimiento de las relaciones entre ellos.
- La definición de la clase Provenance.
- La definición de la clase Policy, así como la definición del algoritmo de asignación de permisos.
- La definición de la clase Packaging.
- La definición de un mecanismo común de empaquetado y entrega de información VO: el VOPack.

Para el futuro, pretendemos:

- Implementar el RADAMS (Radio Astronomy DAta Model for Single-dish, multiple-feed telescopes) para la antena DSS63.

- Contribuir a IVOA con el RADAMS, presentándolo como una IVOA Note para el grupo de interés en Radio astronomía.
- Contribuir al Grupo de Trabajo sobre la Capa de Acceso a Datos de IVOA con el VOPack.
- Extender el RADAMS más allá de DSS63, con el objetivo final de poder utilizarlo para la antena IRAM 30m, que resulta ser el telescopio en longitudes de onda milimétricas más importante del mundo.

Appendix A

Archive Structure and Implementation Proposal

Figure A.1 illustrates the structure we propose for the archive, with a clear layer separation for the different subsystems.

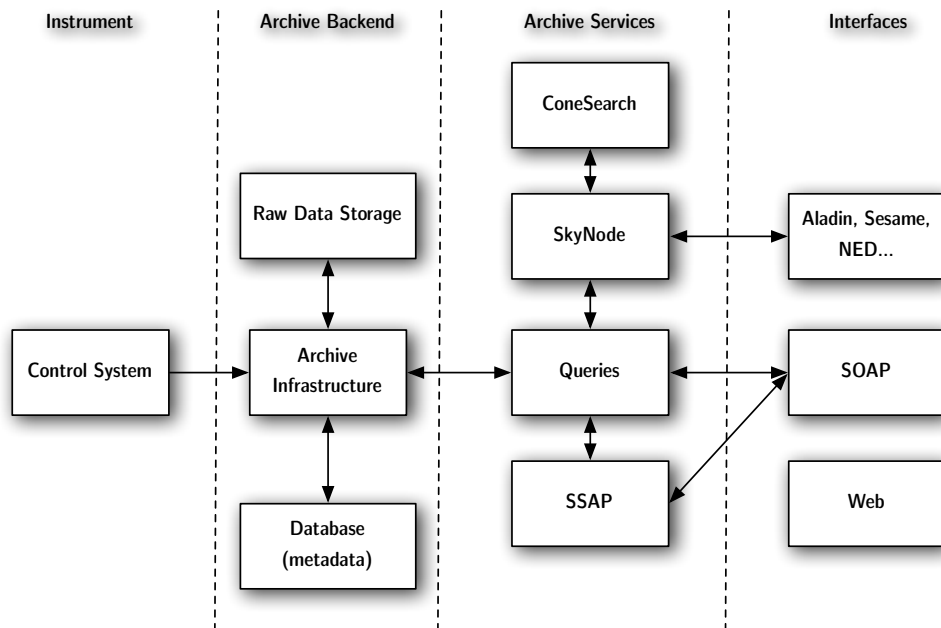


Figure A.1: High level, layered architecture for the Robledo Archive. Dotted lines represent the logical separation between layers. Arrows represent data flow between sub-systems. Communications between layers are confined to the communications established between common components.

We have divided responsibilities in four layers: **Instrument**, **Backend**,

Archive Services and Interfaces:

- The **Instrument**, with regards to the archive, is nothing else but the Instrument Control System, which will provide us with observational data and configuration metadata.
- The **Archive Backend** (not to be confused with any of the instrument's back-ends) is responsible for database and metadata access and maintenance.
- The **Archive Services** layer allows access to the archive to VO clients (software packages and services). In this layer is where we implement machine-to-machine interaction.
- The **Interfaces** layer allows human interaction with the Archive Services, or any other external services we might wish to replicate.

A.1. Instrument Control System

The Instrument Control System is by definition instrument-dependant, and changes to it will not be required. However, we will monitor the activity of the Control System, so that we can learn when new data are available for incorporation into the archive. Of course, instrument engineers might wish to upgrade the Control System in order to better integrate it with the archive.

A.2. Archive Backend

The Archive Backend comprehends all the software packages involved with Control System interaction —so that we can react to Control System's notifications, or to changes in the file system—, with archive database population, with semi-automated metadata generation, and information access and recovery.

For the Backend we will need:

- A relational database where data will be recorded. We will select it among widely available products, and which use, at least, the complete SQL92 instruction set. The database will support transactions and validations, so that transactions maintain relational integrity, and changes to table belonging to the same transaction either succeed or fail together. It should provide also triggers, so that erasing/marking procedures can be conveniently tracked, validated, and logged. XML support is not mandatory, but desirable. Possible candidates are MySQL, PostgreSQL, Oracle, DB2, and others.

- Interaction with the Control System, either via file system monitoring, or explicit procedure call for database population.
- Connection to the Internet, for access to external VO services (Sesame, NED, etc), and to provide SOAP services for external tools.
- A data storage system for the database and final service products (FITS, VOTables, etc). We will not store files in the DB, but instead we will store pointers to files, for later retrieval. We will try to use as much as possible of the pre-existing infrastructure. In fact, for the Robledo Archive its parent organization, INTA, will provide this infrastructure, which will be hosted in their premises.
- Automated or semi-automated archive cataloguing and archive storage software; for each new FITS, we will explore its headers, in order to derive the maximum amount of metadata from them; additional metadata will be gathered from additional sources —observation logs, control system logs— if available, and the user will always be able to add or correct existing metadata.
- Automated logging software, for system audit. We will need to devise a set of users, and database access profiles.

A.3. Archive Services

The Archive Services will be built both for external access and as a basis for the Interfaces layer. These services will be built with platform neutral languages, and will allow automated coupling of database and interface, so that database updates are translated with ease into changes to the Interfaces, either by automatic code generation tools, or dynamic systems with introspection.

In order to do this, we will study web and web-services frameworks such as Tomcat and Axis (Java), Ruby on Rails (Ruby), WASP (PHP), Django (Python), and similar systems.

Archive Services will be built using standard SOAP web services, described by the Web Services Description Language (WSDL).

The service with the highest priority for the Robledo Archive will be the SSAP (Simple Spectra Access Protocol) [17], that will allow for direct machine-to-machine discovery of spectra available for a given sky region.

A.4. Interfaces

Visual interfaces will be of two kinds: standard browser interfaces (thin clients), and desktop clients (thick clients). Thick clients will not be developed in this first phase of the archive, but will be built on top of the existing Archive Services functionality.

Browser interfaces will also make use of this Archive Services, and will present standard XHTML (XML-based HyperText Markup Language) content, with CSS (Cascading Style Sheets) for formatting. This way, we will also be able to use XSLT transformations to create XHTML from the XML information provided by the database system.

Interactivity will make use of modern techniques, such as AJAX (Asynchronous JavaScript And XML), in order to provide online visualization tools.

Appendix B

Archive Workflow

The archive architecture tell us how to divide the archive functionality in several more manageable, modular units, and how does information flow between them. However, that architecture comprises only the elements needed to retrieve already stored data from the archive. We still need to establish how will information be incorporated into the archive, and that is what we call Archive Workflow.

In order for a particular set of FITS files to be incorporated to the archive, the following actions will have to be performed :

1. File selection.
2. Automatic metadata generation.
3. Web-based interface for metadata edition.
4. Data query and XML and VOTable serialization.

The following sections detail this steps.

B.1. File selection

The first step is the selection of the files to be incorporated to the archive. Initially, we will only support FITS files. We will either provide a command-line or a web-based tool for file upload, with the ability to recursively scan folders in order to incorporate files.

The web-based tool will allow a further refinement of the selection, to exclude some of the initially scanned files.

B.2. Automatic metadata generation

For the finally selected FITS files, all of their headers will be read, and the following information derived from either FITS headers, FITS HDUs, or *a priori* knowledge of the telescope and instruments.

- **ObsData:** Observation data will be directly retrieved from the FITS HDU.
- **Characterisation:** Characterisation data will be partially derived from FITS headers, while some other parts of the metadata will need *a priori* knowledge of the telescope or instruments. In particular:
 - **Coverage.Location** can be retrieved from the FITS headers for all AxisFrames.
 - **Coverage.Bounds** can be calculated from the FITS headers for AxisFrame.Spatial and AxisFrame.Temporal; it can be calculated from the **ObsData** for AxisFrame.Observable; and it can be calculated from the FITS headers and knowledge of the instrument for AxisFrame.Spectral.
 - **Coverage.Support** can be calculated from the FITS headers for AxisFrame.Spatial and AxisFrame.Temporal; it can be calculated from the **ObsData** for AxisFrame.Observable; and it can be calculated from the FITS headers and knowledge of the instrument for AxisFrame.Spectral.
 - **Coverage.Sensitivity** needs *a priori* knowledge of the telescope and instruments, apart from the data in the FITS headers or HDUs.
- **Provenance** metadata cannot be directly retrieved from FITS header information; however, it is possible to build sensible defaults for the Provenance metadata from the FITS headers, regarding some of the calibrations being performed, and antenna settings, while user input will be necessary for the remaining metadata.
- **Target** metadata can be retrieved from FITS header information AND an additional Target database; with that information sensible values will be provided for confirmation or alteration by the user.
- **Packaging** metadata will be built from the selection of files conforming an archive entry. These files will be related by belonging to the same VOPack.
- **Policy** metadata can be derived from default policies, together with the FITS header information.

- **Curation** metadata will have sensible defaults for a given telescope-instrument-curator triplet. Telescope and instrument can be obtained from the FITS header, and the Curator can be obtained from the logged in user. However, Curation metadata will remain editable.

B.3. Web-based interface for metadata edition

After all metadata have been calculated or set to sensible default values, the user will be shown a web-based interface that will allow him to edit the assigned values, including defaults that were not applicable, or entering metadata not available from elsewhere.

In any case, the interface will be organised so that an executive summary shows most commonly altered values, and the remaining metadata will be available through different tabs, corresponding to the different metadata classes.

B.4. Data query and XML and VOTable serialization

After metadata editing, all pieces of metadata are available, and both raw data and metadata will be entered in the database, enabling data queries.

The XML classes, VOTables, or VOPacks will be created on the fly from user requests and the data available in the database. Data queries will be built from a web-based interface to the archive, or by means of web-services' requests.

Appendix C

Policy Determination

We will use Policy, Users and ObsData metadata in order to select the corresponding role for the agent just logged in. Figure C.1 shows the flow diagram for the role selection.

This could be easily changed into a role-enabling algorithm that enables different roles for the same user, and displays all the different roles the user can access. If this is not needed, we will stick to the proposed algorithm.

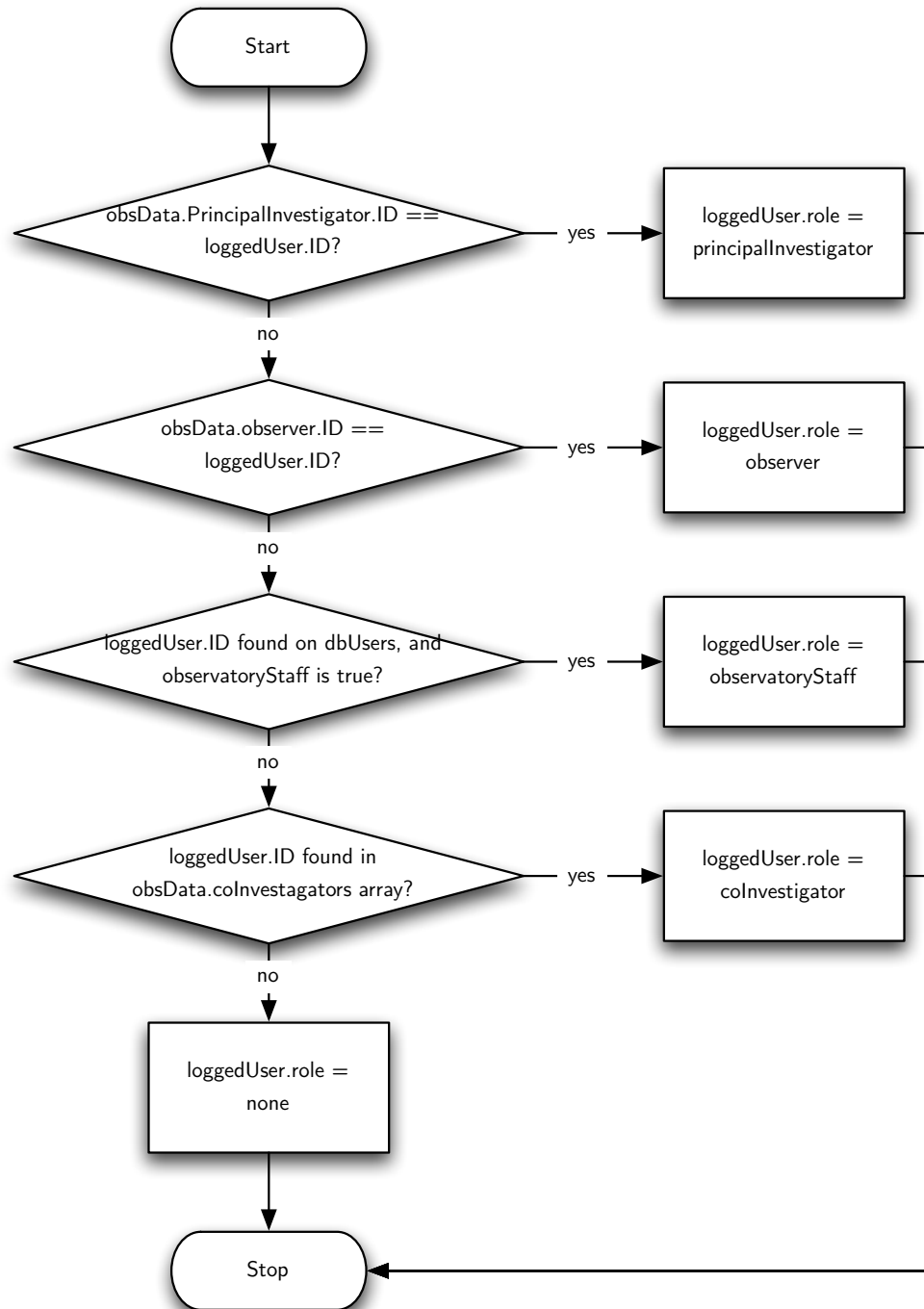


Figure C.1: Flow diagram for the role determination algorithm.

Appendix D

VOPack

The VOPack is a way of distributing VO-compliant content, in a way that makes it easy to reuse and point to existing content, either remote or locally.

A VOPack consists of a compressed file that contains at least a `voPack.xml` file—following the VOPack XML Schema—that describes all the additional content of the compressed VOPack, and their relationships between them. Figure D.1 shows the structure diagram of a VOPack.

In that diagram, the `voPack` element is the root for the XML document. It includes a description, the originating query, and one or more `packUnits`, which actually point to the information being retrieved. The `originatingQuery` element contains the string with the URI that allows the retrieval of the `voPack`. Additional `characterisation` elements, following the Characterisation schema, can be used to further specify properties on the data being delivered with the VOPack.

The `packUnit` corresponds to a single piece of data, or to another `packUnits`, in case of more structured data. The depth of inclusion is arbitrary.

`packUnits` have a `type` attribute that can be one of:

- `votable`
- `fits`
- `otherXML`
- `otherNonXML`
- `vopack`
- `compressedFolder`
- `folder`

For the last three types, a new `vopack.xml` file has to be provided for their description. This allows for meta-packaging of ready-made VOPacks.

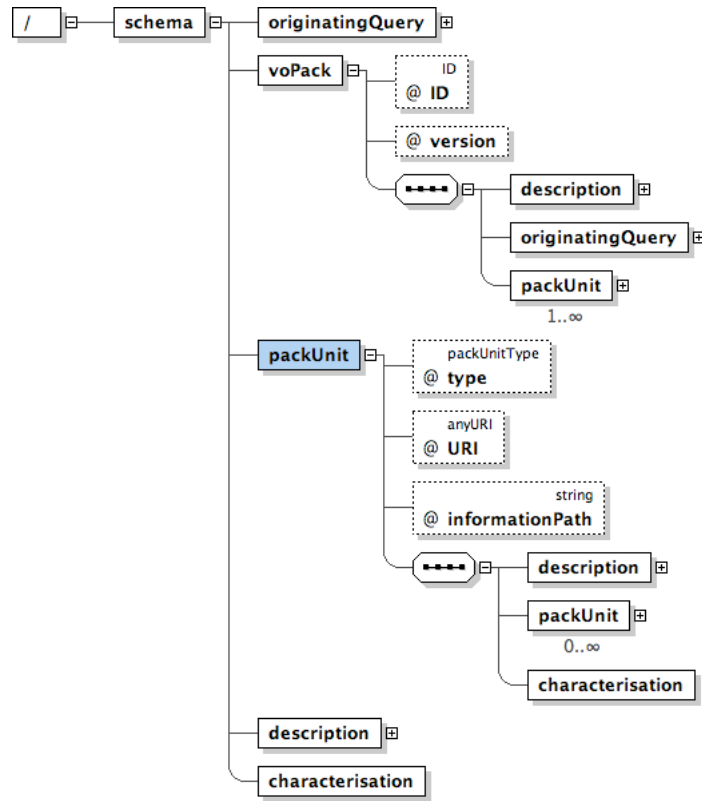


Figure D.1: VOPack structure. Diagram generated by **Oxygen** from the XML schema.

For the first three types, the `informationPath` attribute gives an XPath to the actual data being pointed, just in case the `packUnit` contains several tables, and not all of them are to be considered. In the case of FITS files, the `informationPath` looks XPath-like, but points to the HDU or Image holding the data.

Figure D.2 shows the complete listing for the VOPack XML Schema.

The VOPack XML Schema has been inspired by the concepts of Digital Items, Digital Item Containers, and Digital Item Components from MPEG-21 [4].

```

<?xml version="1.0" encoding="UTF-8"?>
<!-- voPack schema, by Juan de Dios Santander Vela, IAA-CSIC -->
<!-- Ideas for voPach schema

    voPack as a list of packUnits; packUnit is an abstract type
    packUnits: fits, votable, folder, compressedFile, voxml, non-voxml, other
    voxml types: sed, characterisation, stc, stcoords
-->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:import namespace="http://www.w3.org/2001/XMLSchema" /></xs:import>
  <xs:complexType name="anyText" mixed="true">
    <xs:sequence>
      <xs:any minOccurs="0" maxOccurs="unbounded" processContents="skip"/>
    </xs:sequence>
  </xs:complexType>

  <xs:simpleType name="packUnitType">
    <xs:restriction base="xs:NMTOKEN">
      <xs:enumeration value="votable"/>
      <xs:enumeration value="fits"/>
      <xs:enumeration value="otherXML"/>
      <xs:enumeration value="otherNonXML"/>
      <xs:enumeration value="vopack"/>
      <xs:enumeration value="compressedFolder"/>
      <xs:enumeration value="folder"/>
    </xs:restriction>
  </xs:simpleType>

  <!-- voPack is the root element of the schema -->
  <xs:element name="voPack">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="description" minOccurs="1" />
        <xs:element ref="originatingQuery" minOccurs="1" maxOccurs="1"/>
        <xs:element ref="packUnit" minOccurs="1" maxOccurs="unbounded"/></xs:element>
      </xs:sequence>
      <xs:attribute name="ID" type="xs:ID"/>
      <xs:attribute name="version" use="required">
        <xs:simpleType>
          <xs:restriction base="xs:NMTOKEN">
            <xs:enumeration value="0.1"/>
          </xs:restriction>
        </xs:simpleType>
      </xs:attribute>
    </xs:complexType>
  </xs:element>

  <xs:element name="description">
    <xs:complexType mixed="true">
      <xs:complexContent>
        <xs:extension base="anyText">
          <xs:attribute name="ID" type="xs:ID" />
          <xs:attribute name="value" type="xs:string" use="required" />
        </xs:extension>
      </xs:complexContent>
    </xs:complexType>
  </xs:element>

  <xs:element name="originatingQuery">
    <xs:complexType>
      <xs:attribute name="value" type="xs:string" use="required"/>
      <xs:attribute name="URL" type="xs:anyURI" use="required"/>
    </xs:complexType>
  </xs:element>

  <xs:element name="characterisation">
    <xs:complexType>
      <!-- To be defined; should follow the characterisation schema -->
      <xs:sequence>
        <!-- Sequence of axis characterisations for the given pack unit -->
      </xs:sequence>
    </xs:complexType>
  </xs:element>

  <xs:element name="packUnit">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="description" minOccurs="1" />
        <xs:element ref="packUnit" minOccurs="0" maxOccurs="unbounded"/>
        <xs:element ref="characterisation" minOccurs="1" maxOccurs="1"/>
      </xs:sequence>
      <xs:attribute name="type" type="packUnitType" use="required"/>
      <xs:attribute name="URI" type="xs:anyURI" use="required" />
      <xs:attribute name="informationPath" type="xs:string" use="optional"/></xs:attribute>
      <!-- The URI provides information either for the local file system, or remotely -->
    </xs:complexType>
  </xs:element>
</xs:schema>

```

Figure D.2: VOPack XSD schema listing.

Bibliography

- [1] T. Murphy, P. Lamb, C. Owen, and M. Marquarding, “Data storage, processing and visualisation for the ATCA,” *ArXiv Astrophysics e-prints*, Enero 2006.
- [2] P. Warner, “NOAO Science Archive - Domain Model,” tech. rep., National Optical Astronomy Observatory, 2004.
- [3] J. M. Martínez, “Mpeg-7 overview,” October 2004.
- [4] J. Bormans and K. Hill, “Mpeg-21 overview v.5,” October 2002.
- [5] J. McDowell, F. Bonnarel, D. Giaretta, G. Lemson, M. Louys, and A. Micol, “Data Model for Observation,” *IVOA Data Model WG Internal Draft*, May 2004.
- [6] J. McDowell, F. Bonnarel, I. Chilingarian, M. Louys, A. Micol, and A. Richards, “Data Model for Astronomical DataSet Characterisation,” *IVOA Note*, p. 40, May 2006.
- [7] J. McDowell, D. Tody, T. Budavari, M. Dolensky, F. Valdés, P. Protopapas, and A. Rots, “IVOA Spectral Data Model,” *IVOA Data Access Layer WG Working Draft*, May 2006.
- [8] P. Lamb and R. Power, “IVOA Data model for raw radio telescope data,” *IVOA Radio Astronomy Interest Group Note for Discussion*, October 2003.
- [9] I. de Gregorio Monsalvo, *Radio Astronomical Study of the Physical Conditions, Kinematics, and Chemistry of the Environment Surrounding Low-Mass Young Stellar Objects*. PhD thesis, Universidad Autónoma de Madrid, Facultad de Ciencias Físicas, Departamento de Física Teórica, May 2006.
- [10] R. J. Hanisch, W. D. Pence, B. M. Schlesinger, A. Farris, E. W. Greisen, P. J. Teuben, R. W. Thompson, and A. Warnock, “Definition of the Flexible Image Transport System (FITS),” Tech. Rep. NOST 100-2.0, NASA/Science Office of Standards and Technology (NOST), NASA Goddard Space Flight Center, Greenbelt MD 20771, USA, 1999.

- [11] D. Muders, E. Polehampton, and J. Hatchell, “Multi-Beam FITS Raw Data Format,” tech. rep., Max-Planck-Institut für Radioastronomie, December 2005.
- [12] R. M. Prestage and M. H. Clark, “Device and Log FITS Files for the GBT,” tech. rep., NRAO Green Bank, December 2004.
- [13] A. Preite Martínez, S. Derriere, N. Gray, R. Mann, J. McDowell, T. McGlynn, F. Ochsenbein, P. Osuna, G. Rixon, and R. Williams, “The UCD1+ controlled vocabulary,” *IVOA Semantics WG Recommendation*, December 2005.
- [14] J. Schwarz and R. Heald, “Software glossary,” Tech. Rep. Version 0.2, Atacama Large Millimeter Array, May 2003.
- [15] A. Rots, “Space-Time Coordinate metadata for the Virtual Observatory,” *IVOA Proposed Recommendation*, March 2005.
- [16] R. Hanisch, G. Greene, A. Linde, R. Plante, A. M. S. Richards, E. Auden, K. T. Noddle, and W. O’Mullane, “Resource metadata for the Virtual Observatory,” in *ASP Conf. Ser. 314: Astronomical Data Analysis Software and Systems (ADASS) XIII* (F. Ochsenbein, M. G. Allen, and D. Egret, eds.), pp. 273–+, 2004.
- [17] M. Dolensky, D. Tody, T. Budavari, I. Busko, J. McDowell, P. Osuna, and F. Valdés, “Simple Spectral Access Protocol,” *IVOA Data Access Layer WG Working Draft*, May 2006.